ELÉIA GISELE MUELLER

AGENTES MEDIADORES

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre em Informática, ao Departamento de Informática da Universidade Federal do Paraná.

Orientadora:
Prof.ª Dr.ª Aurora Trinidad Ramirez Pozo

CURITIBA
2001

Ministério da Educação
Universidade Federal do Paraná
Mestrado em Informática

PARECER

Nós, abaixo assinados, membros da Banca Examinadora da defesa de Dissertação de Mestrado em Informática da aluna ***Eléia Gisele Mueller***, avaliamos o trabalho intitulado "***Agentes Mediadores***", cuja defesa foi realizada no dia 29 de março de 2001. Após a avaliação, decidimos pela aprovação da candidata.

Curitiba, 29 de março de 2001.

Profª. Dra. Aurora Trinidad Ramirez Pozo
Presidente - Orientadora

Prof. Dr. Jorge Muniz Barreto
Membro Externo - UFSC

Prof. Dr. Martin Alejandro Musicante
DINF/UFPR

Universidade Federal do Paraná
Departamento de Informática
Mestrado em Informática

Dissertação de Mestrado

Agentes Mediadores

Mestranda
Eléia Gisele Mueller

Orientadora
Dra. Aurora Trinidad Ramirez Pozo

Curitiba, março de 2001.

Eléia Gisele Mueller

Agentes Mediadores

Este exemplar corresponde à redação final da dissertação apresentada, como requisito parcial para a obtenção do grau de Mestre em Informática, ao Departamento de Informática da Universidade Federal do Paraná, sob a orientação da Prof. Dra. Aurora Trinidad Ramirez Pozo.

Curitiba – Pr
2001

Dedicatória

Dedico esta conquista aos meus pais, que sempre me apoiaram na realização dos meus sonhos.

Agradecimentos

Primeiramente, agradeço a DEUS que nos criou, nos deu o dom da inteligência e sempre está ao nosso lado, no cumprimento de nossa jornada, mostrando-nos uma luz nas situações mais difíceis.

Agradeço aos meus pais pela confiança que depositam em mim, pelo incentivo que sempre me deram, principalmente nos momentos difíceis da minha vida. Mas agradeço principalmente, por nunca terem medido esforços na educação de seus filhos, nos estimulando na realização de nossos sonhos.

Um obrigado muito especial para a professora Aurora T. R. Pozo, pela sua dedicação e esforço para a realização deste trabalho, onde não houve limites para cumprir com o seu papel de orientadora. Meu obrigado por todas as vezes que sentou ao meu lado e dispôs do seu tempo para ajudar-me a contornar alguma dificuldade. Obrigado pela confiança de que depositou em mim, ao crer que eu chegaria ao final desta caminhada. Sem dúvida, sua compreensão e estímulo ajudaram muito para que eu continuasse a andar. Isto fez com que eu tivesse uma grande admiração e carinho por você.

Meus infinitos agradecimentos ao amigo Fábio M. Lopes, por ter sido muito mais que um colega de mestrado. Obrigado pelo seu companheirismo.

À amiga Simone Vieira meu obrigado, pelo laço de amizade criado e fortalecido durante este período. Gostaria de agradecer ainda, os bons momentos fora deste curso e pela sua alegria que sempre contagiou a todos.

A você Grasiela Drozino que sempre se mostrou amiga, mesmo quando a distância nos separava.

Aos meus tios Inês e Almeida que me acolheram como uma filha quando vim para Curitiba, muito obrigado. Vocês mostraram ser grandes pessoas, dignos da minha admiração e carinho.

Aos demais colegas que encontrei em Curitiba, seja através do mestrado ou não, agradeço pela amizade que nos uniu, une e espero que sempre una e pela atenção que sempre dedicaram à minha pessoa.

E não poderia deixar de agradecer você Marcos, que nestes últimos meses se tornou parte da minha vida. Sua compreensão e suas palavras de conforto e incentivo me fizeram caminhar com mais segurança. Seu amor me mostrou um outro lado da vida, que até então eu não conhecia.

Obrigada a CAPES pelo apoio financeiro. Sem este, dificilmente teria conseguido concluir este curso. A Allegro Common Lisp por ceder uma versão de teste para o desenvolvimento do experimento prático deste trabalho.

Sumário

LISTA DE ABREVIATURAS ..... **VIII**

LISTA DE FIGURAS ..... **IX**

LISTA DE TABELAS ..... **X**

RESUMO ..... **X**

RESUMO ..... **XI**

ABSTRACT ..... **XII**

CAPÍTULO 1 - INTRODUÇÃO **1**

1.1 CONTEXTUALIZAÇÃO DO TRABALHO 1
1.2 OBJETIVOS DA DISSERTAÇÃO 4
1.3 ESTUDO DE CASO 4
1.4 MOTIVAÇÃO PARA O DESENVOLVIMENTO DESTA DISSERTAÇÃO E CONTRIBUIÇÕES DA MESMA 6
1.5 ORGANIZAÇÃO DESTA DISSERTAÇÃO 7

CAPÍTULO 2 - MEDIADORES **9**

2.1 INTRODUÇÃO 9
2.2 O CONCEITO DE MEDIADOR 10
2.3 O DOMÍNIO DE CONHECIMENTO 11
2.4 TRABALHO RELACIONADO: 12
2.5 DISCUSSÃO SOBRE MEDIADORES 14

CAPÍTULO 3 - AGENTES **15**

3.1 INTRODUÇÃO 15
3.2 IAD E AGENTES 15
3.3 CONCEITOS DO TERMO "AGENTE" 16
3.4 CLASSIFICAÇÃO DOS AGENTES 18
3.5 PROPRIEDADES DE UM AGENTE: 22
3.5 AGENTE COGNITIVO X RACIOCÍNIO E SEU DESENVOLVIMENTO 25

CAPÍTULO 4 - DESENVOLVIMENTO DE SISTEMAS BASEADO NO PARADIGMA 'AGENTE' 28

4.1 INTRODUÇÃO 28
4.2 PARÂMETROS PARA UMA METODOLOGIA DE MODELAGEM DO CONHECIMENTO E METODOLOGIAS EXISTENTES PARA O DESENVOLVIMENTO DE AGENTES: 28
4.2 REUTILIZAÇÃO E MODELOS GENÉRICOS DE AGENTES 32
4.3 FERRAMENTAS PARA O DESENVOLVIMENTO DE SISTEMAS BASEADOS NO PARADIGMA AGENTE 34
4.3.1 LOOM *37*

CAPÍTULO 5 - ESTUDO DE CASO: ACOMED **41**

5.1 INTRODUÇÃO 41
5.2 VISÃO GERAL DO ACOMED 42
5.3 PROJETO DO ACOMED 44
5.4 FLUXO DE RACIOCÍNIO DO ACOMED 57

CAPÍTULO 6 - IMPLEMENTAÇÃO DO ACOMED **60**

6.1 INTRODUÇÃO 60

6.2 IMPLEMENTAÇÃO DO MUNDO 60
6.3 IMPLEMENTAÇÃO DAS CRENÇAS DO ACOMED 63
6.4 IMPLEMENTAÇÃO DOS DESEJOS DO ACOMED 64
6.5 IMPLEMENTAÇÃO DOS COMPROMISSOS 66
6.6 IMPLEMENTAÇÃO DO GERENCIAMENTO DO MUNDO 68
6.7 IMPLEMENTAÇÃO DO GERENCIAMENTO DE AGENTES 69
6.8 IMPLEMENTAÇÃO DAS TAREFAS ESPECÍFICAS DO ACOMED 69
6.9 RESULTADOS DA IMPLEMENTAÇÃO 70

CAPÍTULO 7 - CONCLUSÃO 74

REFERÊNCIAS BIBLIOGRÁFICAS: 78

Lista de Abreviaturas

ACOMED:	Agente Cognitivo Mediador
Agente-M:	Agente Matriculador
BD:	Base de Dados
BDI:	Beliefs, Desires and Intentions (Arquitetura de Agentes baseada em crenças, desejos e intenções)
DESIRE:	Design and Specification on Interaction Reasoning (ferramenta para o projeto e especificação de componentes racionais)
IA:	Inteligência Artificial
IAD:	Inteligência Artificial Distribuída
LOOM:	Linguagem de Representação do Conhecimento
SAGU:	Sistema de Apoio ao Gerenciamento Universitário
SISMAT:	Sistema de Matricula Inteligente
SMA:	Sistema Multi-Agente
UFPR:	Universidade Federal do Paraná

Lista de Figuras

Figura 1.1 – Ambiente de desenvolvimento do Estudo de Caso 5
Figura 2.1 – Mediador 9
Figura 2.2 – Diagrama Resumo SIMS 13
Figura 4.3 – Exemplo Loom 39
Figura 5.1 – Visão Geral do Estudo de Caso 42
Figura 5.2 – Refinamento do Controle do Próprio Processo do ACOMED 45
Figura 5.3 - Crença interna do ACOMED 46
Figura 5.4 – Crença do ACOMED baseado no fato comunicado por outro agente 47
Figura 5.5 – Crença do ACOMED baseada na observação do mundo 47
Figura 5.6 – Determinação de desejo do ACOMED 48
Figura 5.7 – Determinação dos desejos do ACOMED 49
Figura 5.8 – Determinação do compromisso, baseado em um desejo do agente 50
Figura 5.9 – Especificação do compromisso baseado em uma meta atingida 51
Figura 5.10 – Especificação da interação do ACOMED com o mundo 52
Figura 5.11 – Meta-conhecimento do Componente Gerenciamento do Mundo 52
Figura 5.12 – Especificação de interação do ACOMED com o agente usuário 53
Figura 5.13 – Meta-conhecimento do componente gerenciamento de Agentes 54
Figura 5.14 – Tarefas Específicas do ACOMED 55
Figura 5.16 – Meta Conhecimento do subcomponente gerar alteração professor 57
Figura 6.2 – Instanciação do conceito aluno 62
Figura 6.3 – Restrição sobre a instanciação do conceito 'day' 63
Figura 6.4 – Implementação da observação do mundo e comunicação da existência da instância procurada 64
Figura 6.5 – Determinação do desejo 66
Figura 6.6 – Determinação do compromisso 67
Figura 6.7 – Definição das tarefas para atingir a meta 67
Figura 6.8 – Gerenciamento do mundo 68
Figura 6.9 – Comunicação do código da informação a ser localizada 69
Figura 6.10 – Implementação da tarefa específica do ACOMED 70
Figura 6.11 – Agente consulta e a chamada ao ACOMED 70

Figura 6.12 – Apresentação em tela do desejo do ACOMED 71
Figura 6.13 – ACOMED comunica que o código da informação existe e pertence a um aluno 71
Figura 6.14 - ACOMED comunica que o código da informação não existe e que o processo está sendo abortado 71
Figura 6.15 – Comunicação do compromisso assumido pelo ACOMED 72
Figura 6.16 – Comunicação de que está se traçando o plano das tarefas a serem executadas 72
Figura 6.17 – Resultado da consulta aos dados de um aluno 72

Lista de Tabelas

Tabela 3.1 – Classificação dos Sistemas de Agentes 29

Resumo

Dentre os vários ramos de estudo em IA que buscam representar o comportamento humano, um deles é de particular interesse nessa dissertação: a Inteligência Artificial Distribuída (IAD), cuja definição, é um refinamento das definições clássicas de IA. A saber: "IAD é o estudo do comportamento computacionalmente inteligente, resultante da interação de múltiplas entidades dotadas de certo grau de autonomia. Estas entidades são usualmente chamadas de **agentes** e o sistema como um todo, de **sociedade**".

Nesta dissertação, enfocaremos o estudo a agentes inteligentes, estando nosso interesse voltado para o tipo mediador. A motivação para a realização deste veio através das dificuldades encontradas pelo grupo de pesquisa da UFPR sobre agentes, no acesso a bases de informações, em trabalhos realizados antes deste (SISMAT e AGENTE-M).

Dentre outros, veremos alguns conceitos de agente inteligente, suas principais características e uma arquitetura geral para o mesmo, que foi proposta por Francis Brazier *at al.* Tal arquitetura está baseada nos estados mentais (crença, desejo e intenção).

Neste sentido, este trabalho vem contribuir com os estudos voltados a compreensão do comportamento humano através da tentativa da simulação do mesmo, juntando os conceitos "agente inteligente", "mediação" e "estado mentais", estando porém, muito longe de representá-lo por completo, devido a sua complexidade.

Vamos expor ainda, o estudo de caso realizado. Tal estudo trata da implementação, através da ferramenta para desenvolvimento de sistemas baseado em conhecimento - LOOM, de um protótipo de um agente mediador cognitivo. Para a definição do agente foi utilizada a arquitetura de Brazier, adaptada para o AGENTE-M. Com isto, conseguimos avaliá-la e verificar que tal arquitetura pode ser reutilizada para a definição de diferentes tipos de agente, com diferentes propósitos.

Abstract

Among the several branches in studying AI that try to represent the human behavior, one of them holds a particular interest: the Distributed Artificial Intelligente (DAI), which is a refinement of AI classical definitions. Notes: "DAI is the intelligent computationally behavior study resulting from the multiple entity interaction which has a certain autonomy level. These entities are commonly named as agents and the whole system is named as society."

The intelligent agent study is presented in this dissertation, and our interest turns to the mediator type. The difficulties found by the UFPR researchers on agents in database access in the earlier works (SISMAT and AGENT-M) was the motivation to its realization.

Some intelligence agent concepts, their main characteristics and a general architecture proposed by Francis Brazier at al are presented. Such architecture is based on mental states (beliefs, wishes and intention).

This work contributes to the studies about comprehending human behavior through his experimental simulation – linking the concepts " intelligent agent", "mediation" and "mental states" – being, however, far away from complete representation due to human complexity.

This work shows a case study which treats the prototype cognitive mediator agent implementation by the system development tool based in knowledge - LOOM. The Brazier´s architecture was used to the agent definition, adapted to the AGENT-M. With that, we could evaluate and make sure that such architecture can be reutilized in the definition of different agent types with different purposes.

Capítulo 1 - Introdução

1.1 Contextualização do Trabalho

Um dos principais objetivos da Inteligência Artificial (IA) é a representação do comportamento humano através de modelos computacionais. A pesquisa em IA enfrenta dois desafios: como armazenar o conhecimento humano e como fazer com que o computador aprenda como um humano.

Dentre os vários ramos de estudo em IA que buscam estes objetivos, um deles é de particular interesse nessa monografia: a Inteligência Artificial Distribuída (IAD), cuja definição, é um refinamento das definições clássicas de IA. A saber:

- "IAD é o estudo do comportamento computacionalmente inteligente, resultante da interação de múltiplas entidades dotadas de certo grau de autonomia. Estas entidades são usualmente chamadas de **agentes** e o sistema como um todo, de **sociedade**"[OLI96].
- " IAD é um campo do conhecimento que estuda e tenta construir conjuntos de entidades autônomas e inteligentes que cooperam para desenvolver um trabalho e se comunicam por meio de mecanismos baseados no envio e recepção de mensagens [QUI99].

Podemos observar que as mesmas dificuldades que surgem quando se tentamos definir IA (como por exemplo a definição do termo inteligência), reaparecem no caso de IAD.

A IAD utiliza um modelo de inteligência baseado no comportamento social, com ênfase nas ações e interações de *'agentes'* que podem ser entidades reais ou virtuais imersas num ambiente, sobre o qual são capazes de agir [ALV97].

Assim, o termo "**agente**" é uma noção central e fundamental para este trabalho. Tal termo vem sendo utilizado para denotar simples processos de hardware e/ou software, até entidades sofisticadas capazes de realizar tarefas complexas. Esta diversidade reflete o

estado atual da área, onde temos diversas definições do que é realmente um agente. Muitos grupos de pesquisa seguem uma determinada linha, apresentando a sua definição personalizada para o termo em questão, de acordo com seus próprios objetivos. Podemos encontrar a definição de agentes em: [HEI95], [RIB98], [QUI99], [RUS95], [FER91], [WOO95], [SOU97], [RIV96], [COE96], [VIR95], [HAY99], [HEI95].

Assim, foi necessário um extenso levantamento das definições do termo agente (apresentadas no capítulo 2), sendo que resumimos o mesmo como um processo composto por humano, *software, hardware* ou por uma combinação destes, capaz de realizar uma determinada tarefa e disponibilizar seus serviços a uma coletividade de outros agentes, que em conjunto, tentarão atingir um objetivo comum. Baseando em [BRA99b], resumimos os objetivos principais de um agente da seguinte forma:

- Realizar tarefas para as quais foi especificado;
- Cooperar com os demais agentes do ambiente, de forma a fornecer as informações e/ou prestar serviços para que estes possam realizar suas tarefas de forma mais eficiente.

Esta definição de agentes é complementada mediante a descrição de um conjunto de propriedades usualmente aceitas pela comunidade de IAD e que os mesmos devem exibir, tais como: autonomia, aprendizagem, racionalidade e adaptabilidade. Detalharemos cada uma no capítulo 3.

Alguns desafios são inerentes para esta área de estudo (IAD). Um dos mais relevantes consiste em coordenar o comportamento inteligente de um conjunto de agentes autônomos (sistemas multi-agentes), os quais devem cumprir um certo objetivo. É fácil perceber a dificuldade desta tarefa. Mesmo no mundo real, organizar um conjunto de humanos "inteligentes" é tarefa árdua e por sinal, deixada a cargo de gerentes especializados. Além disto, em se tratando de informática, os agentes irão competir pelos mesmos recursos limitados, tais como: tempo, espaço, *software* e *hardware*.

Portanto, o objetivo principal de um sistema multi-agentes, em coordenar o comportamento inteligente de um conjunto de agentes autônomos, é beneficiar-se dos resultados de outros agentes (pertencentes ao sistema) ou ajudá-los, por razão de eficiência e a fim de evitar conflitos entre eles, para obter a solução do problema apresentado.

Sendo a IAD, um campo de pesquisa 'interdisciplinar', ela vem ultimamente, tentando resolver um outro ponto crítico que enfrentamos: o grande número de bases de informações (bases de dados, programas, arquivos *flat*, etc.) armazenadas e disponíveis para consultas, sendo que estas bases além de serem heterogêneas, na maioria dos casos, ainda estão distribuídas e podem estar armazenadas em linguagens diferentes. Logo, torna-se árduo o trabalho de busca de uma determinada informação, em tais bases. Assim surge a tecnologia dos **agentes mediadores** dentro de IAD.

Um **'agente mediador'** é um agente responsável em buscar uma informação necessária para outra entidade, sendo que este agente deve saber selecionar corretamente a base onde se localiza tal informação. Esta outra entidade, que pode ser entendida como outro agente (*softwares*, *hardwares* e até mesmo um humano), possui diversos termos diferentes, que forma uma ontologia, própria de cada um. Embora um termo possa ser sinônimo de outro (mesma semântica), na maioria dos casos possuem sintaxes diferentes. Logo o agente mediador deve também saber decodificar o pedido da informação (*consulta*), isto é, deve antes de tudo, traduzi-la para a sua linguagem padrão, especializando a *consulta* recebida dentro de uma 'ontologia' entendida por ele.

Para facilitar o desenvolvimento de um agente com as características citadas acima, é preciso selecionar uma metodologia para desenvolvimento de sistemas baseados em conhecimento, uma arquitetura geral para tal agente, que irá auxiliá-lo nos seus raciocínios e no retorno da informação desejada. Também é necessário escolher uma linguagem com alto poder computacional para implementá-lo, onde seja possível representar todas as suas características.

Neste projeto, estamos especificamente interessados neste tipo de agente, o mediador, e em seus aspectos cognitivos. Para tal projeto, foi selecionado o DESIRE [BRA99a], [BRA99b], [BRA98a], [BRA98b], [TRE99] como metodologia base para o desenvolvimento de sistemas cognitivos (inteligentes) e sua arquitetura para agentes, que também foi base para o Agente-M: Um matriculador Inteligente [GUI00]. Para a definição dos estados mentais do agente mediador, seguimos o mesmo espectro do agente matriculador implementado em [GUI00].

A tarefa do agente mediador neste projeto consiste em acessar uma base de dados e retornar uma informação (dado), caso ela esteja presente nesta base. Selecionamos o banco de informações da Universidade Federal do Paraná (UFPR), como referência para o estudo de caso que será explicado na próxima sessão, construindo assim, uma base espelho para os nossos estudos. Por motivos claros (complexidade e tamanho), apenas modelamos uma parte desta base.

Nosso trabalho propõe-se a verificar a viabilidade do uso de agentes mediadores para acesso a base de dados e quais são as características inerentes a ele. Assim, concluindo o mesmo, proporemos um modelo para um agente mediador cognitivo. Tal modelo tem como pretensão contribuir com o avanço nas pesquisas em IAD, sobretudo quando se tratam de agentes inteligentes mediadores.

1.2 Objetivos da dissertação

São objetivos específicos da dissertação:

- Estudar os aspectos (positivos e negativos) de um módulo mediador entre aplicações de usuários e bases de dados;
- Estudar os conceitos, as propriedades e os aspectos cognitivos de agentes inteligentes, centrando-se nos 'agentes mediadores';
- Implementar um agente mediador inteligente utilizando a ferramenta para desenvolvimento de sistemas baseados em conhecimento, LOOM [LOOM], enfocando os seus aspectos cognitivos.
- Baseando-se no trabalho Agente-M: Um matriculador Inteligente, discutir os aspectos de reutilização do modelo genérico para agentes proposto pelo DESIRE.

1.3 Estudo de Caso

Ao falarmos nas tarefas dos agentes, destacamos a de um agente atuar como mediador entre bases de dados (distribuídas ou não) e outras entidades (que neste caso

devem ser compreendidas como aplicações de qualquer natureza, isto é, outros agentes). Estes agentes devem localizar a base de dado (BD) correta, caso ela esteja distribuída e nesta, encontrar a informação solicitada pela outra entidade, se a informação for consistente.

Aplicações deste tipo vem sendo motivada principalmente pelo tamanho das BD's, que aumentam naturalmente com o passar do tempo. Com isto, não há necessidade de modificar constantemente os sistemas desenvolvidos e nem que os mesmos conheçam as BD's completamente e suas localizações, sendo que ambas estão sujeitas a mudanças. Assim, podemos ter um ambiente como o ilustrado na Figura 1.1.

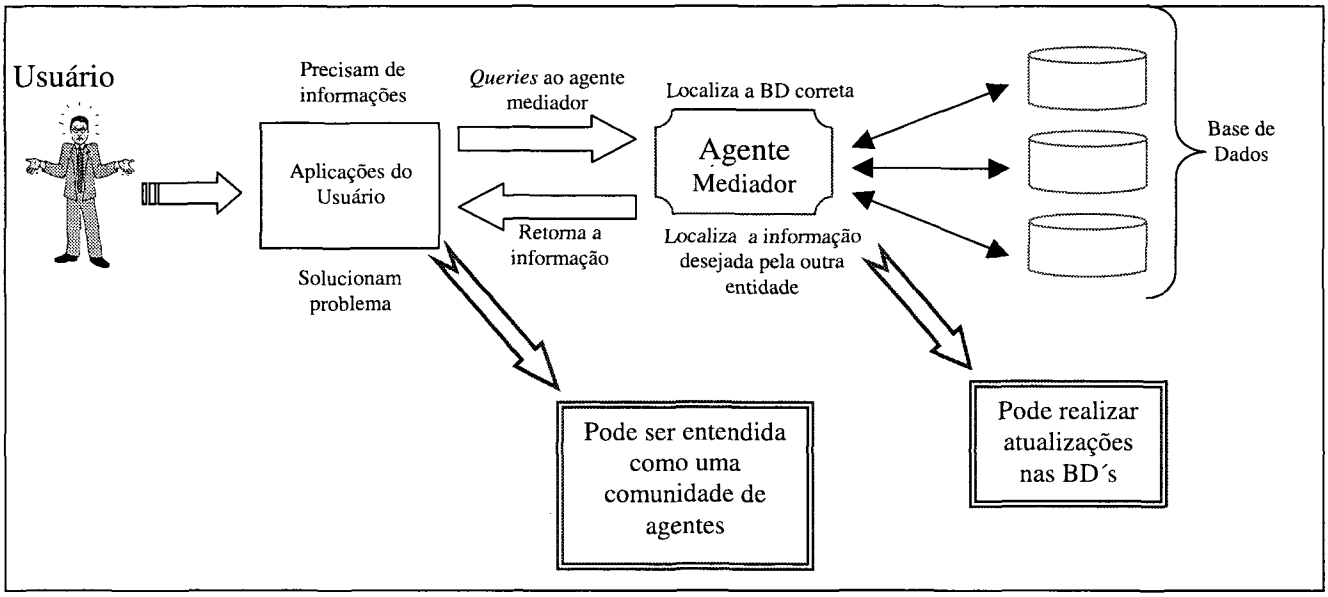

Figura 1.1: Ambiente de Desenvolvimento do Estudo de Caso

O estudo de caso escolhido será a implementação de um agente mediador simulando as bases de dados da Universidade Federal do Paraná (UFPR) e o sistema de informação, SAGU (Sistema de Apoio ao Gerenciamento Universitário), em desenvolvimento na mesma, que faz parte do projeto "Sistemas de Suporte a Decisão" [SAGU].

Dentro deste projeto, temos diversas aplicações que necessitam de informações. Este agente mediador deve receber alguns parâmetros, vindos do usuário ou diretamente das aplicações, verificar se é possível obter tal informação e em caso positivo, sair em busca

desta, nas bases de dados existentes. Num próximo passo, o agente mediador deve retornar tal informação ao agente solicitante.

1.4 Motivação para o desenvolvimento desta dissertação e contribuições da mesma

As aplicações industriais, que são uma das categorias de problemas que devemos solucionar através de aplicações computacionais, são geralmente complexas. Complexidade demanda decomposição do problema, afim de solucioná-lo mais facilmente. Por isso, esta é uma das áreas aonde a cooperação entre agentes autônomos vem sendo aplicada com sucesso, surgindo assim os sistemas multi-agentes (SMA).

A decomposição de um problema complexo demanda coordenação e comunicação entre os agentes responsáveis pela solução do mesmo. A cooperação entre os 'resolvedores' de problemas deve-se estruturar como uma série de intercâmbios de informação, cuidadosamente planejados, que obviamente dependerão do problema.

Os resultados apresentados até o presente momento pelas pesquisas realizadas pelo grupo de pesquisa do Departamento de Informática da UFPR sobre "agentes cognitivos" estimulam a concretização desta dissertação, pois cremos que os avanços dentro desta abordagem, mais especificamente na abordagem mediador, melhorarão a qualidade da obtenção de informações armazenadas em grandes bases, já que mediadores processam conhecimento, juntamente com regras e operações.

Como resultado destas pesquisas, podemos citar outras duas dissertações: SISMAT Sistema de Matricula Inteligente [MAT00] e Agente-M: Um Matriculador Inteligente [GUI00], onde vemos que uma das dificuldades destes trabalhos é a obtenção das informações necessárias para realizar uma matricula. Isto nos motivou a realizar o estudo de caso descrito na seção anterior.

Ainda temos como fonte motivadora deste trabalho a verificação da característica 'genérico' do modelo DESIRE, isto é, a reutilização deste modelo para diferentes aplicações de agente e quais são as vantagens de utilizá-lo.

1.5 Organização desta dissertação

Além da presente introdução, dividimos esta dissertação em mais outros seis capítulos, afim de que se obtenha uma melhor compreensão do trabalho realizado. São eles:

- ✓ **Capítulo 2 – Mediadores**: Neste capítulo vamos abrir uma discussão sobre mediação: definição, o papel de um módulo mediador e trabalhos relacionados com mediação.
- ✓ **Capítulo 3 – Agentes**: Neste capítulo colocamos os resultados obtidos na tentativa de conceituar o termo agente. Além disto, descrevemos as principais propriedades de um agente e suas principais classificações, dando destaque aos agentes cognitivos, em particular os agentes BDI.
- ✓ **Capítulo 4 – Desenvolvimento de Sistemas Baseados em Agentes**: Neste capítulo vamos abordar os aspectos de desenvolvimento de um sistema baseado em agentes. Iremos descrever a arquitetura proposta por Brazier *et al* [BRA98, BRA99] para desenvolvimento de sistemas sob este paradigma; abriremos uma seção para falarmos sobre as ferramentas de implementação para agentes, onde destacaremos o LOOM – ferramenta utilizada no nosso estudo de caso. Ainda falaremos sobre outros trabalhos realizados na área de agentes, em especial os trabalhos realizados pelo grupo de pesquisa sobre agentes cognitivos da UFPR.
- ✓ **Capítulo 5 – Estudo de Caso**: Aqui será explicada detalhadamente a implementação feita em LOOM para acesso a base de dados, ou seja, iremos explicar como foi definido o agente cognitivo mediador através da adaptação do modelo DESIRE e do Agente-M: Um Matriculador Inteligente.
- ✓ **Capítulo 6 – Implementação do ACOMED**: Este capítulo fala como foi implementado cada processo definido no agente mediador, além de explicar como foi implementado o mundo (base de informações) do agente.

- ✓ **Capítulo 7 – Conclusões**: Finalizando este trabalho, vamos colocar as conclusões obtidas no decorrer da realização desta dissertação e os trabalhos futuros.

Capítulo 2 - Mediadores

2.1 Introdução

Com a atual explosão de dados, retornar e integrar informações de várias bases torna-se um problema crítico. Para contornar esta situação e otimizar o tempo de procura de uma informação é necessário conhecimento sobre as bases e sobre a forma de como os dados estão armazenados.

O conceito de mediadores, tal como proposto por Wiederhold [WIE92], aponta para esta direção ao propor uma classe de software capaz de lidar com a informação de forma ativa e inteligente. Com este objetivo define-se uma camada responsável pelo *link* entre aplicações de usuários e os bancos de dados, provendo a integração informacional e sua relevância (Figura 2.1).

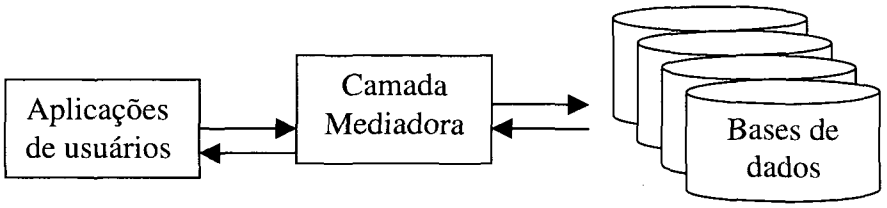

Figura 2.1 – Mediador

Os mediadores no conceito de Wiederhold [WIE92] incorporam conhecimento técnico e administrativo para produzir informações relevantes requisitadas por módulos que atendem, principalmente, usuários. Neste capítulo vamos fazer uma breve discussão sobre o conceito de 'mediador'.

2.2 O Conceito de Mediador

A medida em que o volume de dados, automação de acesso e número de usuários aumenta, as funções de mediação inteligente se tornam mais necessárias. Mediar, em termos gerais consiste na capacidade de lidar com linguagens de interlocutores de mundos distintos, que têm algo a buscar e oferecer uns para os outros, mas necessitam de intermediação para que possam se entender [FLE97].

Segundo Wiederhold [WIE92], mediadores são módulos que ocupam uma camada ativa entre as aplicações e os recursos de dados. Um mediador é um módulo de software que utiliza conhecimento codificado sobre certos conjuntos ou subconjuntos de dados (domínio) para criar informação para uma camada mais alta das aplicações. Assim, os mediadores podem oferecer acesso em um dado domínio da aplicação, para informações que residem em bases normalmente heterogêneas e distribuídas.

O mesmo Wiederhold enfatiza que não há necessidade de um mediador oferecer a um usuário uma interface de fácil uso, pois esta função cabe as suas aplicações. Ele deve oferecer uma interface que seja fácil para o processamento da comunicação.

Mediadores oferecem serviços intermediários, ligando bases de dados e aplicações. Sua função é oferecer a informação integrada, sem a necessidade de integrar as bases de dados. Especificamente, as tarefas requeridas para cumprir esta função são [WIE95]:

1. Acesso e retorno de dados relevantes armazenados em múltiplas bases heterogêneas;
2. Abstração e transformação dos dados retornados, de modo que eles possam ser integrados;
3. Integração dos dados homogeneizados de acordo com suas chaves;
4. Processamento dos dados integrados para aumentar a densidade da informação no resultado;

Vamos na próxima seção discutir sobre o domínio onde o um mediador, mais especificamente, um agente mediador, deve buscar as informações.

2.3 O Domínio de Conhecimento

As questões relacionadas com a manutenção de sistemas baseados em conhecimento no passado não foram adequadamente analisadas: manutenibilidade de grandes bases de conhecimento demonstrou-se vital para que tais sistemas perdurem e se expandam [WIE90].

O conhecimento foi tratado como algo relativamente estático e por isso os requisitos relativos à sua manutenção não foram devidamente atendidos. Embora algum conhecimento seja estável por muito tempo, novos conceitos acabam por surgir, requerendo manutenção das bases de conhecimento. A importância e a extensão destas manutenções sugerem que os sistemas sejam voltados para domínios específicos que favoreçam pequenas e/ou bem delimitadas unidades de conhecimento.

A especialização de mediadores aumenta seu poder e manutenibilidade. Além disso, o conhecimento incorporado aos mediadores não pode ser estático, ao contrário, é baseado em realimentação.

O conhecimento pode ser adquirido e modificado através de especialistas humanos Verificação de inconsistências entre dados e conhecimento acompanha sempre o processo de atualização das bases de conhecimento. A atualização de um conhecimento já existente, com novos parâmetros advindos de mudanças nos dados, em geral é simples. A aquisição de novos conceitos oferece maior dificuldade e normalmente requer intervenção de um especialista humano [FLE97].

2.4 Trabalho Relacionado:

Esta seção oferece o resumo sobre um trabalho realizado antes do nosso e que por utilizar mediação, serve como base para esta dissertação. Tal trabalho utiliza o conceito de mediador como uma aplicação capaz de acessar bases heterogêneas e distribuídas e retornar as informações necessárias para outras aplicações.

2.4.1 - SIMS

O SIMS [SIMS] é um sistema desenvolvido pelo grupo de pesquisa em IA da Universidade da Califórnia do Sul, utilizando a linguagem de conhecimento Loom [LOOM]. Ele destina-se a resolução de problemas derivados da tentativa de oferecer acesso a múltiplas bases de dados e conhecimento.

Especificamente, o SIMS é responsável pelo seguinte [ARE92]:

- Determinar qual base de informação contém o dado relevante para a base de conhecimento usada na formulação de uma consulta;
- Para todas as classes mencionadas em uma consulta e que parecerem não ter base de informação relacionada, determinar se algum conhecimento encoberto no modelo do domínio (tal como relações para outras classes) permite reformulação de maneira que habilitará bases de informação para serem identificadas.
- Criar um plano, uma seqüência de sub-consultas e outras formas de manipulação de dados que quando executadas produzem a informação desejada;
- Usar o conhecimento sobre as bases de dados para otimizar o plano;
- Oferecer uma maneira uniforme para descrever as bases de informações para o sistema, além dos dados que estão acessíveis nelas.

A Figura 1.2 apresenta um esquema de representação visual do SIMS.

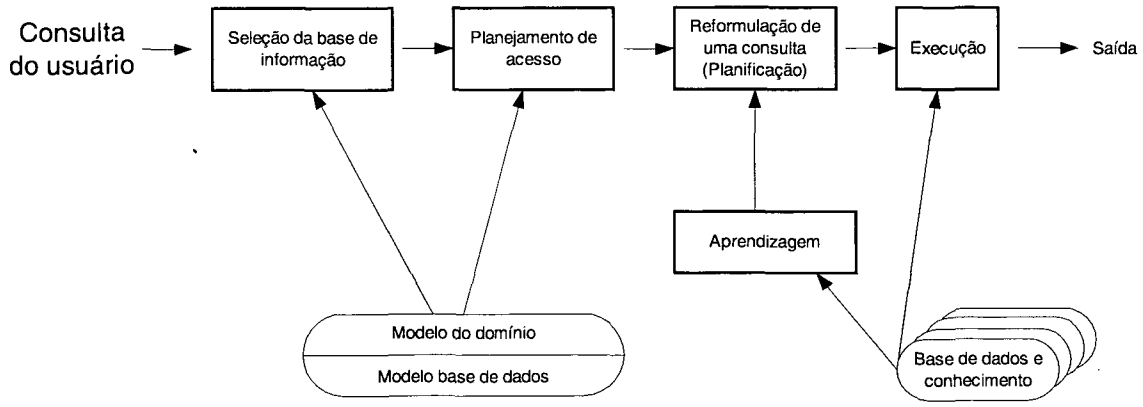

Figura 2.2 – Diagrama Resumo SIMS

O SIMS deve raciocinar sobre dados e outros conhecimentos armazenados em uma variedade de locações e formatos. É de extrema importância que ele tenha descrições detalhadas das várias bases de informações na qual ele tenha acesso – nenhum sistema pode retornar a informação requisitada se ele não tem conhecimento sobre onde a informação em questão pode estar armazenada e como acessá-la.

No SIMS um modelo de cada base de informação é criado para descrevê-la ao sistema. Além disso, um modelo do domínio é construído para descrever objetos e ações que são significantes no desempenho das tarefas do domínio da aplicação. A coleção de termos do modelo do domínio forma o vocabulário usado para caracterizar o conteúdo de uma base de informação.

É importante notar que a modelagem das diferentes bases de informações são independentes umas das outras. Isto simplifica a tarefa de modelagem ao mesmo tempo em que habilita novos componentes para serem adicionados ao SIMS.

Modelagem das Bases de Informações:

Para cada base de informação, o 'modelo SIMS' deve incluir todos os fatos que podem influenciar nas decisões, concentrando em quando utilizá-las. O primeiro passo para responder uma consulta expressa em termos do modelo do domínio é selecionar a base de

informação apropriada. Isto é feito no SIMS pelo mapeamento do conceito no modelo do domínio para os conceitos no modelo das bases de dados que correspondem diretamente às bases de informações.

2.5 - Discussão sobre mediadores

Como visto neste capítulo, os mediadores são uma camada de *software* entre bases de dados e aplicações de usuários. Tal camada de software tem como missão oferecer o acesso eficaz a estas múltiplas bases que podem ser heterogêneas ou não.

A maioria dos trabalhos feita dentro deste contexto até a atualidade trata os mediadores como um sistema baseado em conhecimento. Porém um novo paradigma tem se mostrado muito forte: o paradigma de agentes.

Assim, este trabalho vem propor uma nova característica aos agentes mediadores: os estados mentais, fazendo com que eles tenham uma certa "inteligência" no acesso as bases e na devolução da informação requisitada.

Os mediadores não são mais tratados como simples sistemas baseados em conhecimento, mas como agentes capazes de comunicar-se com os demais agentes do ambiente e de realizar seu trabalho eficientemente.

Através da integração do conceito de agentes e do conceito de mediadores, podemos ver que um agente mediador pode desempenhar funções vitais, como busca de informações, coordenadas por sistemas de crenças, valores e métricas de confiabilidade, poder autônomo de decisão e ações pautadas por um código de ética, de acordo com os requisitos típicos de uma arquitetura de agentes [FLE97].

Portanto faz-se necessário um estudo aprofundado sobre agentes e sobre o seu desenvolvimento. Estudo esse, que será discutido nos capítulos 3 e 4. Devido à complexidade do tema, apenas implementamos para fim de estudo, um protótipo de um mediador, apresentado nos capítulos 5 e 6.

Capítulo 3 - Agentes

3.1 Introdução

Neste capítulo colocaremos os principais pontos sobre agentes. O surgimento desta tecnologia trouxe consigo uma variedade de definições sobre o referido termo. Em suas propostas de definição para agente, podemos observar que os pesquisadores, na maioria dos casos, tentam explicar o enfoque da sua pesquisa, o que acaba criando novos significados para o termo "agente". Isto nos levou a uma pesquisa abrangente sobre estas definições e suas propriedades. Com esta pesquisa, desejamos verificar onde nossa aplicação se insere e quais propriedades nos serão úteis.

3.2 IAD e Agentes

A IA clássica (origem psicológica) tem seu estudo centrado no comportamento individual do ser humano. Os pesquisadores buscam respostas para a melhoria dos seus sistemas no modelo de inteligência do indivíduo, utilizando os diversos métodos de representação do conhecimento e de inferência.

A IAD (origem sociológica) utiliza um modelo de inteligência baseado no comportamento social, com ênfase nas ações e interações de agentes que podem ser entidades reais ou virtuais imersas num ambiente sobre o qual são capazes de agir [ALV97].

O termo agente é uma noção central e fundamental para área de IAD. Este termo vem sendo utilizado para denotar desde simples processos de *hardware* e/ou *software* até entidades sofisticadas capazes de realizar tarefas complexas. Esta diversidade reflete o estado atual da área, onde vemos diversos conceitos do que é realmente um agente.

3.3 Conceitos do termo "agente"

Esta seção tem por objetivo apresentar as definições adotadas por diversos grupos de pesquisa, para o termo 'agente', além de apresentar as duas principais classificações aceitas para agentes: cognitivos e refletivos.

As definições destes conceitos e a apresentação das classificações fazem-se necessárias a fim de posicionar o presente trabalho dentro da área de IAD. Nas definições encontradas sobre agentes, podemos observar a diversificação das idéias, e ver como elas diferem entre si, conforme o contexto onde estão inseridos. A seguir apresentamos alguns dos conceitos encontrados sobre 'agente':

- Alguém ou algo que age como representante de uma outra parte, para a finalidade de realizar atos específicos que parecem ser benéfica à parte representada [HEI95].
- Designa uma entidade inteligente e autônoma. A palavra autônoma, neste caso, significa que cada agente possui sua própria existência, a qual não é dependente da existência de outros agentes [RIB98].
- Uma entidade física ou abstrata que pode perceber seu ambiente através de sensores, capaz de identificar tais percepções e tomar decisões por meio de mecanismos de raciocínio simples ou complexos, comunicar-se com outros agentes para obter informações e atuar sobre o meio em que se desenvolve através de executores [QUI99].
- Um agente é qualquer coisa que pode ser vista como perceptora do ambiente (por sensores) e capaz de atuar nele (por "efetores") [RUS95].
- Agente é uma entidade real ou virtual que está imersa em um ambiente onde pode realizar ações, que é capaz de perceber e representar parcialmente este ambiente, de comunicar-se com outros agentes e que possui um comportamento autônomo, que é consequência de suas observações, seu conhecimento e interações com outros agentes [FER91].

- Agente é uma entidade que possui e explicitamente representa, modelos simbólicos do mundo e no qual as decisões são feitas por meio de raciocínio simbólico [WOO95].
- Agentes inteligentes são entidades de *software* que realizam algum conjunto de operações em benefício do usuário ou de outro programa, utilizando um certo grau de independência ou de autonomia e ao fazê-lo, empregam algum conhecimento ou representação dos objetivos ou preferência do usuário [SOU97].
- Um agente é um programa de computador que funciona em *"background"* e desenvolve tarefas autônomas conforme as delegadas pelo usuário [RIV96].
- Agentes são programas que travam diálogos, negociam e coordenam transferência de informações [COE96].
- Os agentes apresentam conceitos de habilidade para execução autônoma e habilidade para executar raciocínio orientado ao domínio [VIR95].
- Agente é uma entidade (computador ou humano) que é capaz de atingir metas e é parte de uma grande comunidade de agentes que têm influência mútua uma sobre o outro [HAY99].
- Agente pode ser definido como alguém ou alguma coisa que atua como um representante para outra entidade, com o propósito expresso de desempenhar ações que são benéficas para a parte representada. É diferenciado de outras aplicações por suas dimensões e capacidade de interagir independentemente da presença do usuário [HEI95].

Assim, chegamos a seguinte conclusão sobre o termo agente: **um processo composto por seres humanos, *software, hardware* ou por uma combinação destes, capaz de realizar uma determinada tarefa e disponibilizar seus serviços a uma coletividade de outros agentes, que em conjunto, irão realizar um objetivo comum.**

Baseado em [BRA99b], resumimos os principais objetivos de um agente da seguinte forma:

- Realizar tarefas para as quais foi especificado;

- Cooperar com os demais agentes da rede, de forma a fornecer as informações e/ou prestar serviços para que os outros agentes possam realizar suas tarefas de forma mais eficiente.

Assim, a definição de '**agente mediador**', passa a ser uma especialização do termo agente, onde temos que sua tarefa é buscar informações necessárias a outras entidades (agentes), nas bases existentes, devendo saber localizá-la corretamente, caso esta esteja distribuída e devolver a informação correta para a entidade que fez o pedido (*consulta*).

3.4 Classificação dos agentes

Nesta seção, vamos definir as duas principais classes de agentes: os agentes cognitivos e os agentes reativos, além do agente BDI, que é um caso particular de agente cognitivo. Na Tabela 3.1, mostramos as principais diferenças entre os sistemas de agentes cognitivos e os reativos.

3.4.1 Agente Reativo:

É um agente de baixo nível, que não dispõe de um protocolo nem de uma linguagem de comunicação e cuja única capacidade é responder a estímulos [QUI99]. Não tem memória das ações realizadas no passado e nem previsão da ação a ser tomada no futuro. Geralmente atuam em sociedades, baseando-se em modelos de organização biológica [RIB98].

3.4.2 Agente Cognitivo:

Conforme descrito em [QUI99], um agente cognitivo é aquele que é capaz de efetuar operações complexas, é individualmente inteligente (é um sistema com um certo grau de esperteza, com capacidade de raciocínio sobre sua base de conhecimento), pode comunicar-se com os demais agentes e chegar a um acordo com eles, sobre alguma decisão. Um sistema cognitivo é composto por um pequeno número de agentes cognitivos. Os avanços nos trabalhos de IAD, nos permitem definir as seguintes características de agentes cognitivos [QUI99]:

- ✓ Intencionalidade: Um agente cognitivo é guiado por suas metas, as quais descrevem as situações que são designadas para o agente. Uma intenção é a declaração explícita de suas metas e meios para chegar a elas, e os planos se podem definir como seqüências de ações que levam um agente a obter sua meta.
- ✓ Racionalidade: Um agente cognitivo possui critérios de evolução de ações e de seleções, de tal maneira que o que decida seja em benefício dele (adquirir mais conhecimento, obter respostas, etc.). Ademais, é capaz de justificar suas decisões.
- ✓ Compromisso: Um agente cognitivo cooperativo planifica suas ações por coordenação e negociação com outros agentes. Estes agentes devem cumprir o acordo ao qual se submeteram.
- ✓ Adaptabilidade: Um agente cognitivo é capaz de controlar suas aptidões e comportamento de acordo com as tarefas que deva realizar no sistema.
- ✓ Inteligência: Um agente cognitivo é inteligente se é racional, intencional e adaptável. Um agente inteligente deve ser capaz de operar com êxito em vários ambientes.

SISTEMAS COGNITIVOS	SISTEMAS REATIVOS
Representação explicita do ambiente	Não tem
Pode ter conhecimento do passado	Não tem memória
Agentes Complexos	Funcionamento estímulo/resposta
Poucos agentes	Muitos agentes

Tabela 3.1: Classificação dos Sistemas de Agentes [QUI99].

3.4.2.1 Agente BDI:

Um segmento da pesquisa em IA tem explorado modelos de agentes baseados em crenças, desejos e intenções. As arquiteturas que seguem este paradigma são conhecidas como arquiteturas BDI (*Belief, Desire,* and *Intention*) e são uma especialização dos agentes cognitivos. As idéias básicas da abordagem BDI são descrever o processamento interno do estado de um agente utilizando um conjunto de categorias mentais (crença, desejo e intenções) e definir uma arquitetura de controle através da qual o agente seleciona racionalmente o curso de suas ações. Os conceitos individuais associados a esta categoria são apresentados a seguir sob diferentes perspectivas [GIR99]:

1. Crenças:

As crenças do agente expressam suas expectativas sobre o estado atual do mundo e sobre a chance de uma ação ou conjunto de ações atingir um certo efeito. Nos trabalhos de Wooldridge [WOO95] e Halpern e Moses [HAL92], as crenças são modeladas usando a semântica de mundos possíveis. Nesse sentido, um conjunto de mundos possíveis é associado com cada situação, denotando os mundos que o agente acreditar serem possíveis. Na visão de Bratman [BRA89], crenças são as visões que o agente possui sobre o ambiente em que ele se encontra. Para Corrêa [COR94], crença é um estado mental intencional fundamental para as interações dos agentes, com noção idêntica a de *conhecimento*, cujo conteúdo externo é uma proposição.

Para Shoham [SHO93] crença é representada por um operador modal B, como:

$$B^{t}_{a}\,\varphi,$$

significando que o agente 'a', no tempo 't', acredita na sentença φ.

2. Desejos:

O desejo é uma noção abstrata que especifica as preferências acerca dos estados futuros do mundo ou o curso das ações que o agente possivelmente quer que se verifiquem. Uma questão importante relacionada ao desejo é que ao agente é permitido ter desejos inconsistentes, que ele não precisa acreditar que possam ser alcançados. Ter desejos não significa que o agente agirá para atingí-los. É um dos estados mentais humanos que não tem uma definição precisa, sendo, portanto, usado e explicado de diversas maneiras. Um estado mental é motivador se é um mecanismo ou representação que tende a produzir, modificar ou selecionar ações à luz das crenças [GIR99]. O desejo é intencional e motivador apresentando, as seguintes características [GIR99]:

- Representa uma situação ou um conjunto de situações em que o agente gostaria que o mundo estivesse;
- Pode estar em conflito com as crenças do agente;
- Pode apresentar simultaneamente desejos conflitantes;
- Não causa diretamente as ações, mas pode potencialmente gerar suas ocorrências.

A idéia de desejo como motivador, no sentido de ter uma representação e um mecanismo que gere e selecione objetivos, tem sido pouco desenvolvida e aplicada na construção de agentes inteligentes. Mesmo nos trabalhos em arquiteturas BDI, os desejos são tratados como objetivos. A sua representação como operador modal do tipo DES (a, p) - o agente 'a' tem como desejo a proposição 'p' - apresenta inconvenientes, visto que se q é uma conseqüência lógica de p (ou seja $p \rightarrow q$), então, da semântica dos mundos possíveis decorre DES (a,q). Isto não necessariamente é verdadeiro.

Nós constatamos esse fato, observando a noção psicológica de desejo. Por exemplo: o desejo de que fazer a viagem implica em gastar muito dinheiro, mas não necessariamente implica no desejo de gastar muito dinheiro.

3. Intenções:

Assim como os desejos, as intenções contêm a representação dos estados que o agente quer que se verifiquem. A base para a definição do conceito de intenção está fortemente associada aos trabalhos filosóficos de Michael Bratman [BRA84], [BRA89]. Este autor claramente distingue o conceito de fazer coisas intencionalmente (ação) e possuir intenção de fazê-las (estado mental). Por exemplo [GIR99]:

1- "Escrevo esta proposta". Intencionalmente escrevo esta sentença caracterizando a ação no conceito de intenção;

2- "Escrevi esta proposta para ilustrar uma nova idéia de agentes". A intenção de exemplificar caracteriza o estado da mente no conceito de intenção. Ter 'intenção de' não garante que a ação será realizada, mas apenas que se intenciona realizá-la.

Uma vez definidos estes três estados mentais, os modelos existentes para uma arquitetura BDI estabelecem as relações entre tais estados. Segundo Móra [MOR99], essas relações podem acontecer de duas formas. Primeiro, as intenções adotadas por um agente são um subconjunto consistente dos desejos deste agente. Segundo, as intenções não podem ser conflitantes com crenças dos agentes. Ou seja, o agente não pode possuir intenções que ele acredite sejam impossíveis de satisfazer.

O modelo genérico de agente inteligente que iremos usar no desenvolvimento da dissertação e que será apresentado na seção 4.2.1, está baseado nos agentes BDI, que como podemos observar, encontram-se numa escala mais alta que os reativos, devido a suas características. Assim, os agentes cognitivos nos dão maior poder para a simulação do comportamento humano e solução de atividades complexas, aumentando o seu campo de aplicação.

3.5 Propriedades de um Agente:

Para que uma tarefa ou projeto seja adequado para a aplicação da abordagem de agente, é necessário identificar algumas características comuns dos mesmos e então, relacioná-las com as perspectivas da resolução do problema proposto. Para que o sistema seja considerado uma aplicação da abordagem agente, ele não necessita apresentar todas as propriedades[1], mas algumas delas, como autonomia, são essenciais.

As propriedades disponíveis nos agentes estão diretamente relacionadas à tarefa ou aplicação para a qual o agente é proposto, sendo que o resultado, deve ser uma arquitetura, que deve dar condições ao projetista de adaptar e modelar o seu agente, conforme suas necessidades. A seguir apresentamos algumas destas propriedades, as quais são fundamentadas em [BEL95], [FRA96], [GIR99] e [HEI95] e diferenciam os agentes de outras aplicações:

- **Autonomia:** É a capacidade de tomar decisões (ações) para realizar algumas tarefas e objetivos, sem a interferência do usuário final. Deve haver um elemento de independência no agente. Como se espera de humanos, os agentes tomam nossos interesses e desejos como entrada e saem por si só para fazer as tarefas esperadas.

  Um agente autônomo deve manter uma agenda independente de seu usuário. Isto requer aspectos de ação periódica, execução espontânea, e iniciativa, na qual o agente deve ser capaz de tomar ações preemptivas ou independentes que irão eventualmente beneficiar o usuário.

- **Comunicabilidade:** Os agentes devem acessar informações de outras fontes, sobre o atual estado do ambiente, durante o curso de suas tarefas. Isto requer a habilidade de comunicar-se com os repositórios desta informação. Estes podem ser outros agentes ou bases de dados.

  Obviamente, quando falamos em comunicação, consideramos que ambas as partes devem falar a mesma língua. Ou seja, é necessário definir um protocolo e interfaces de comunicação, de uma forma bem detalhada e precisa.

[1] Nesta seção, estamos tratando propriedade como sinônimo de característica.

- **Mobilidade:** É a capacidade que os agentes podem possuir de mover-se espacialmente buscando ou conduzindo conhecimento, com o objetivo de resolver determinadas tarefas.

- **Confiabilidade:** Deve existir uma forte confiança que o agente vai representar o seu usuário com precisão. Os agentes inteligentes devem demonstrar veracidade, que é a suposição que este não comunicará informações falsas, e benevolência, que é a suposição que ele não terá objetivos conflitantes e não atuará de modo contraditório ao seu objetivo.

- **Aprendizagem**: os agentes devem aprender com as informações oriundas do ambiente ou de outros agentes;

Além destas características citadas, que são imprescindíveis a um agente, é desejável que ele apresente mais algumas, descritas a seguir:

- **Iniciativa:** É a habilidade de exibir comportamento direcionado ao objetivo, oportunístico e que não reage simplesmente ao seu ambiente. A iniciativa está relacionada à reatividade, pois diante de um determinado estímulo, o agente deve reagir tomando uma iniciativa na execução de certas tarefas.

- **Comportamento Adaptativo:** Para manter as capacidades de autonomia e raciocínio, o agente deve ter alguma forma de analisar o estado de seu domínio externo, ou seja, a porção do ambiente ao alcance do agente, e incorporar isto em suas decisões sobre ações futuras. Os agentes devem ser capazes de examinar o seu ambiente externo, e o sucesso de ações anteriores tomadas sob circunstâncias similares e adaptar suas ações com o intuito de aumentar a probabilidade de alcançar com sucesso seus objetivos.

- **Percepção:** Um agente deve perceber o ambiente e as mudanças ocorridas neste.

Os agentes, ditos inteligentes, devem apresentar além das propriedades citadas anteriormente, em especial, as propriedades de raciocínio e sociabilidade. Estas propriedades são discutidas a seguir:

- **Raciocínio:** A capacidade de raciocínio é talvez o aspecto mais importante que distingue um agente inteligente dos outros agentes. Dizer que um agente inteligente tem raciocínio significa dizer que ele tem a habilidade de inferir e induzir, com base no conhecimento atual e experiências, numa maneira racional e reprodutível.

- **Sociabilidade:** Significa "imitar" características inteligentes dos humanos, com o objetivo de atribuir maior grau de inteligência (capacidade de manipular conhecimento) para os agentes, onde estes demonstram conhecimento e segurança para resolução das tarefas para as quais são concebidos.

Nos sistemas multi-agentes, os agentes devem, em especial, ser cooperativos uns com os outros. Esta característica é detalhada a seguir:

- **Cooperação:** Como extensão natural da comunicabilidade, os agentes devem ter um espírito cooperativo. A idéia é que eles trabalhem juntos para realizar tarefas complexas, aproveitando-se das particularidades de cada um para cumprir seus objetivos.

3.5 Agente Cognitivo x Raciocínio e seu desenvolvimento

Os agentes cognitivos freqüentemente exibem uma rica variedade de comportamento refletivo: eles "raciocinam" sobre vários aspectos, não somente sobre seu próprio comportamento, mas também sobre os comportamentos de outros agentes e as interações entre eles, no caso de sistemas multi-agentes. Segundo [BRA99a], [BRA99b] e [BRA98b], mais especificamente um agente deve raciocinar sobre:

- seu próprio estado de informação;
- suas suposições;
- controle de seus raciocínios;
- suas observações;
- sua comunicação;
- outros estados de informação de agentes, suposições, processos de raciocínio, etc.

Assim, faz-se necessário uma metodologia de modelagem de sistemas baseados em conhecimento para a o desenvolvimento de sistemas multi-agentes que possua uma maneira de expressar estes diferentes tipos de raciocínio reflectivo. Em particular, um ambiente que dê poder para modelar o alinhamento dos diferentes tipos de reflexão e adaptá-los conforme a necessidade.

Atualmente, existem várias técnicas para a modelagem de sistemas baseados em conhecimento, fazendo com que estes sejam construídos com menos erros que antes e maior confiabilidade, o que não ocorria antigamente, pois tais sistemas eram construídos de forma "*ad-hoc*", sem uso de técnicas específicas [WER95]. Tais técnicas devem dar a possibilidade de verificarmos quais das características apresentadas na seção 3.3 são relevantes para o sistema que está sendo modelado.

No próximo capítulo, vamos apresentar a metodologia de desenvolvimento de sistemas baseados em conhecimento, DESIRE [BRA98, BRA99] e a arquitetura para agente inteligentes, derivada da mesma, escolhidos para o desenvolvimento desta dissertação. Além

disto, também apresentaremos a linguagem selecionada para a implementação do estudo de caso, LOOM [LOOM].

Capítulo 4 - Desenvolvimento de Sistemas Baseado no Paradigma 'AGENTE'

4.1 - Introdução

Qualidade em software não se atinge de forma espontânea. É necessário ter um processo de desenvolvimento sistemático e adequado, isto é, uma 'engenharia de software' adequada à construção de sistemas. O objetivo de utilizar metodologias de engenharia de software na construção de sistemas baseado em conhecimento é dar apoio ao desenvolvimento, à documentação e à manutenção do mesmo [WER95].

Nesta sessão, vamos abordar a metodologia para desenvolvimento de sistemas baseados em conhecimento, sob o enfoque agente, DESIRE e a arquitetura geral para tal paradigma, obtida desta. Ambos foram propostos por Frances Brazier *et al* [BRA98a, BRA98b, BRA99a, BRA99b]. Tal metodologia foi utilizada no desenvolvimento do estudo de caso.

Também vamos discutir sobre a linguagem para implementação de sistemas baseados em conhecimento, LOOM [LOOM], desenvolvida pela Universidade da Califórnia do Sul e utilizada na implementação do nosso estudo de caso. Além disso, vamos verificar algumas implementações utilizando o paradigma agente.

4.2 Parâmetros para uma metodologia de modelagem do conhecimento e metodologias existentes para o desenvolvimento de agentes:

As metodologias para modelagem de conhecimento definem a organização básica de uma abordagem envolvendo além de conhecimento, raciocínio. Estas metodologias distinguem os tipos básicos de componentes da modelagem, suas relações e propostas para o desenvolvimento do modelo [BRA99].

Tipicamente, cada ferramenta de modelagem do conhecimento é também associada com um número de ferramentas de suporte e linguagens de modelagem e desenvolvimento.

Um ambiente de modelagem do conhecimento deve ser descrito, em termos de três parâmetros principais [BRA98b]:

1. **Tipos de componentes:** Uma descrição dos diferentes tipos de estruturas do modelo.
2. **Relações entre componentes** - Uma descrição das principais relações estruturais entre os componentes da modelagem.
3. **Metodologia de desenvolvimento do modelo** - Uma descrição da metodologia de desenvolvimento do modelo, associado a uma abordagem de modelagem (exemplo: DESIRE - agentes).

Assim como a engenharia de software caminhou para metodologias estruturadas de desenvolvimento de sistemas, hoje em IA, existe uma concentração de esforços buscando igualmente metodologias estruturadas para desenvolvimento de sistemas baseados em conhecimento.

Dentre as existentes na literatura, a escolhida foi a metodologia DESIRE, proposta por Francis Brazier *et al* [BRA98a],[BRA98b],[BRA99a],[BRA99b], por permitir expressar os diferentes tipos de pensamento refletivo que um agente inteligente deve possuir. Esta metodologia será discutida em mais detalhes na próxima seção.

4.1.1 DESIRE – Uma metodologia para o desenvolvimento de sistemas sob o paradigma agente:

A metodologia de modelagem de sistemas multi-agentes DESIRE [BRA98a, BRA98b, BRA99a, BRA99b], permite expressar os diferentes tipos de raciocínio reflexivos existentes e também modelar estes tipos de reflexão, através da decomposição de tarefas. As trocas de informações entre os módulos são especificadas no modelo, através de ligações entre os mesmos, sendo que cada ligação relata a saída de uma informação do componente para entrada em outro.

No DESIRE, uma modelagem baseia-se nos seguintes passos: composição de processos, composição de conhecimento e a relação entre ambos

[BRA98a],[BRA98b],[BRA99a],[BRA99b]. Estes três tipos de conhecimento são discutidos com mais detalhadamente a seguir.

⇒ **Composição de Processo**

A composição de processo identifica os componentes relevantes nos diferentes níveis de abstração e descreve como este pode ser definido (composto) em termos de outros processos (sub-componentes) de níveis mais baixos.

- **Identificação de processos em diferentes níveis de abstração**

  Os processos podem ser descritos em diferentes níveis de abstração: por exemplo, o processo de sistemas multi-agentes como um todo; o processo definido por agentes individuais e o mundo externo; e os processos definidos pelas tarefas individuais dos agentes. Os processos identificados em cada nível de abstração são modelados como componentes do sistema e para cada um, são especificados quais são as informações de entrada e saída.

  Os níveis identificados no processo de abstração são modelados como relação de abstração/ especialização entre os componentes, os quais podem ser formados por outros componentes ou podem ser primitivos. Os componentes primitivos podem ser baseados em uma base de conhecimento, ou componentes capazes de executar tarefas específicas como cálculos.

- **Decomposição de Processos**

A maneira na qual os processos pertencentes a um nível de abstração são compostos por outros processos pertencentes a níveis de abstração adjacentes ou mais baixos é chamada decomposição. Esta é descrita por uma especificação das possibilidades de troca de informações entre os mesmos (visão estática na composição) e a especificação das tarefas que controlam o conhecimento (visão dinâmica na composição).

⇒ **Composição do Conhecimento**

A composição do conhecimento identifica a estrutura do conhecimento em níveis de abstração diferentes e descreve como a mesma pode ser definida em termos de outras estruturas de níveis mais baixos.

- **Identificação das estruturas do conhecimento em diferentes níveis de abstração**

  As duas principais estruturas usadas como blocos de construção para modelar o conhecimento são os tipos de informações e as bases de conhecimento. As estruturas do conhecimento podem ser identificadas e descritas em qualquer nível de abstração. Em níveis mais altos, os detalhes não precisam ser especificados.

  Um tipo de informação define uma ontologia (vocabulário léxico) para descrever objetos ou termos, seus tipos e as relações ou funções que podem ser definidas nestes objetos. Uma base de conhecimentos define uma parte do conhecimento que é usado em um ou mais processos.

- **Composição das estruturas do conhecimento**

  Os tipos de informações podem ser decompostos em tipos de informações mais específicos, seguindo o princípio da decomposição. Similarmente, as bases de conhecimento também podem ser decompostas em bases de conhecimento mais específicas. Assim, a estrutura composicional é baseada nos diferentes níveis de abstração do conhecimento, resultando em informações e conhecimento não especificados.

⇒ **Relação Entre Composição do Processo e Conhecimento**

Cada processo quando é decomposto, usa estruturas do conhecimento. Qual estrutura do conhecimento é usada para qual processo é definida pela relação entre ambos.

4.2 Reutilização e Modelos Genéricos de Agentes

As fases de modelagem de processo e conhecimento freqüentemente consomem muitos recursos e inteligência [BRA98b]. Para limitar o tempo e a especialidade requeridos para desenvolver um sistema baseado em conhecimento, uma metodologia de desenvolvimento para tais sistemas deve reutilizar tantos elementos quanto for possível. Qual modelo deve-se usar, depende da descrição do problema [BRA98b, BRA99a, BRA99b]. As descrições genéricas de agentes e suas tarefas podem ser usadas para gerar uma variedade de agentes específicos e tarefas, descritas através do refinamento e decomposição dos seus processos [BRA98b, BRA99a]. Os modelos genéricos para agentes e suas tarefas podem ser genéricos em dois sentidos:

- com respeito as tarefas (abstração das tarefas do agente em níveis mais baixos);
- com respeito ao conhecimento (abstração do conhecimento em níveis mais baixos, por exemplo, especificar o domínio de uma aplicação).

O refinamento de um modelo genérico para níveis mais baixos, resulta em um modelo mais específico; este processo é chamado *especialização*. O refinamento de um modelo genérico para um nível de abstração mais baixo do conhecimento, por exemplo, modelar um domínio específico de aplicação, é chamado de *instanciação* [BRA98b, BRA99a].

Sistemas composicionais de desenvolvimento para sistemas baseados em conhecimento sob o paradigma agente centram sua atenção em ambos aspectos de genericidade, freqüentemente começando com um modelo genérico do agente. Este modelo pode ser modificado ou refinado através de especializações e instanciações. A aplicabilidade de um modelo genérico de agente depende das características do problema.

4.2.1 Um Modelo Genérico de Agente

As características atribuídas aos agentes variam significativamente, dependendo do problema e das tarefas para os quais eles são designados. Os agentes podem ou não, por

exemplo, ser capazes de comunicar-se com outros agentes. Um agente reflexivo pode somente ser capaz de reagir a informações vindas do mundo externo. Um agente social e cognitivo, em comparação, pode ser capaz de planejar, monitorar e cooperar com outros agentes. Qual modelo é melhor para uma situação é determinado durante a análise do sistema [BRA98b, BRA99a]. Em geral, a solução adotada, corresponde a uma combinação de modelos.

Para a modelagem do um agente genérico, apresentado na Figura 4.2, foram considerados os seguintes aspectos, já discutidos anteriormente: composição de processos, composição de conhecimento e a relação entre eles. Este modelo foi proposto por Frances Brazier *at al*, [BRA98a, BRA98b, BRA99a, BRA99b, TRE99] e suporta a noção de agentes BDI, veja Figura 4.1. Os seis componentes deste modelo são:

- *Controle do Próprio Processo* (OPC): através da decomposição deste componente, temos a determinação das crenças, desejos, intenções e compromissos que o agente pode assumir (Figura 4.2). Este componente é responsável por determinar, planejar agendar e monitorar uma atividade do agente. Além disto, ele é responsável em manter todas as informações relevantes na atividade do agente e o *status* da mesma.

- *Manutenção das Informações do Mundo* (MWI): atualiza e interpreta as informações vindas do mundo externo. Além disso, armazena todas as informações obtidas pelo monitoramento do mundo.

- *Gerenciamento da Interação com o Mundo* (WIM); gerencia a comunicação com o mundo do agente. Ele é responsável pela execução das observações e ações. Uma importante sub-tarefa deste componente, é a observação dos efeitos do processo executado no mundo por outros agentes ou por ele mesmo.

- *Manutenção das Informações dos Agentes* (MAI): Sob solicitação, este componente fornece outro nome de agente ou de subcomponente capaz de executar certa atividade especificada.

- *Gerenciamento da Interação com Agentes* (AIM); gerencia a comunicação com outros agentes, em particular, com o grupo de agentes pertencentes ao seu ambiente. Ele recebe informações vindas destes agentes e as transfere para outros componentes.

- *Tarefas Específicas do Agente* (AST): determina as tarefas do agente, os métodos de aquisição das informações e as suposições necessárias, além de gerenciá-las e validá-las. Também avalia o estado dos seus processos.

Tendo discutido a metodologia de desenvolvimento DESIRE, vamos na próxima seção abordar uma linguagem para a implementação de sistemas baseado em conhecimento, o LOOM.

4.3 Ferramentas para o Desenvolvimento de Sistemas Baseados no Paradigma Agente

Devido às características apresentadas pelos agentes inteligentes, características estas que os diferenciam dos sistemas convencionais, fazem-se necessárias ferramentas mais específicas ao seu desenvolvimento. Como a teoria dos agentes inteligentes é relativamente recente, encontramos poucas ferramentas comerciais para esta tarefa, sendo que a maioria das aplicações existente é acadêmica, oriundas dos centros de pesquisa das universidades.

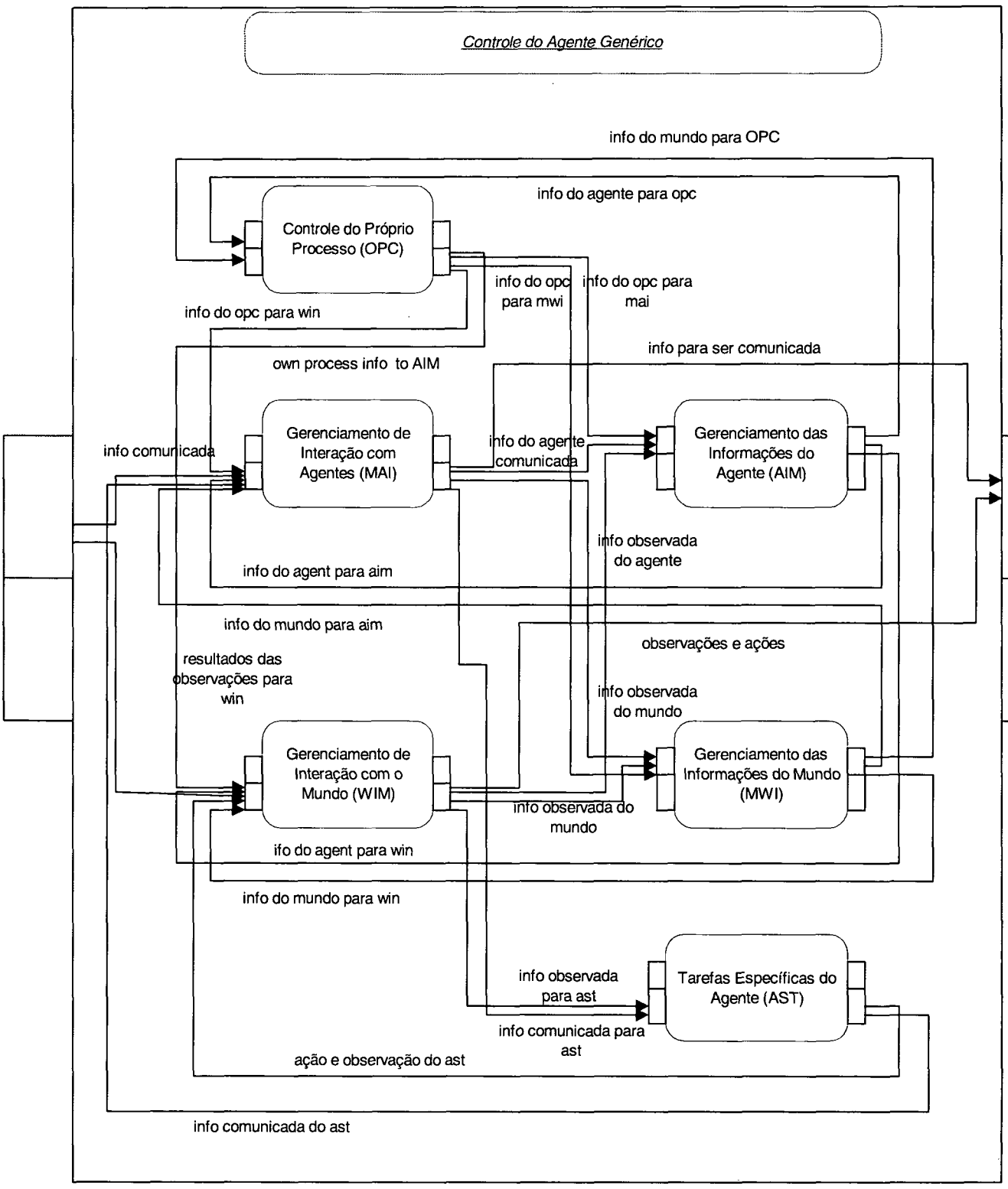

Figura 4.1 – Modelo Genérico Para Agentes [BRA98a, BRA98b, BRA99a, BRA99b, TRE99].

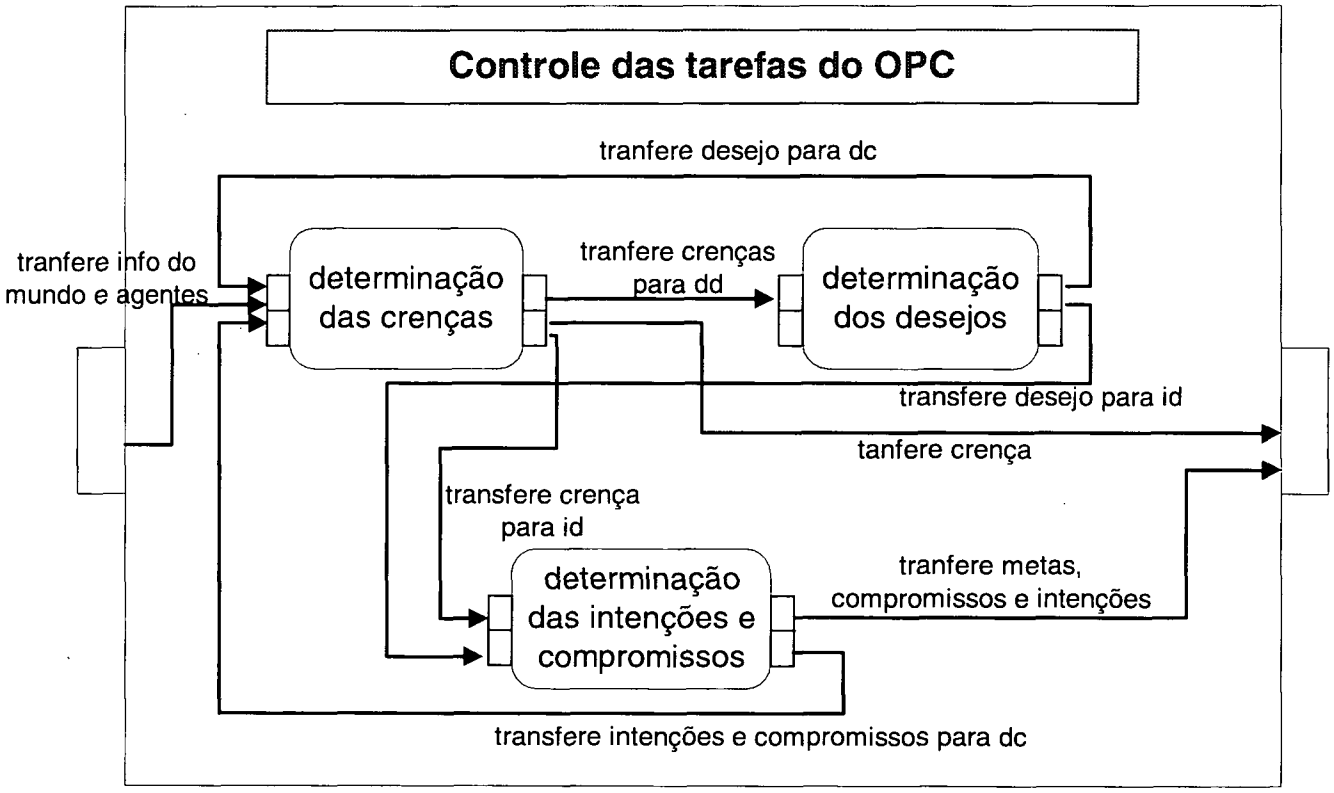

Figura 4.2 – Refinamento do OPC [BRA98a, BRA98b, BRA99a, BRA99b, TRE99].

Existem basicamente três tipos de linguagem para o desenvolvimento de agentes [MAT00]:

1. Linguagens Genéricas: trata-se do desenvolvimento dos sistemas partindo-se praticamente do zero, somente com alguns recursos oferecidos pelas linguagens que serão utilizadas, como C, C++, *Pascal, Prolog* e *Lisp*, através de suas bibliotecas padrões, ou seja, terá que ser implementado todas as características da teoria de agentes;
2. *Frameworks*: neste caso, o desenvolvedor terá a sua disposição, para a implementação do sistema, uma biblioteca específica para a atividade, que cobre os pontos principais, os mais complexos e gerais da teoria de agentes, ficando para a equipe de desenvolvimento a modelagem do agente, que deve visar as tarefas que o mesmo deverá executar;

3. Linguagens específicas: nesta opção a parte relativa ao agente está pronta, sob a forma de um esqueleto de aplicação, que atende a um conjunto específico de áreas de atuação de agentes, como: aprendizagem, pesquisa e processamento de tarefas. Nesta abordagem a modelagem do agente prende-se mais a adaptação do mesmo ao ambiente em uso e a definição específica da finalidade do mesmo.

Atualmente, existem algumas ferramentas específicas para o desenvolvimento de sistemas baseados em conhecimento, especialmente sob o enfoque 'agentes'. Estas ferramentas têm como objetivo principal, facilitar a programação de tais sistemas, através de comandos pré-definidos para executarem tarefas complexas, que exigiriam grandes esforços. Praticamente todas as ferramentas enfocam a comunicação entre agentes e a mobilidade dos mesmos, mas não se preocupam com os aspectos cognitivos.

Dentre as ferramentas para o desenvolvimento de sistemas sob o paradigma agente, escolhemos o 'pacote' denominado LOOM para a implementação do estudo de caso. Esta ferramenta foi escolhida primeiramente, por oferecer a possibilidade de trabalhar-se com diferentes tipos de raciocínio. Também, foi levado em consideração que a mesma é uma biblioteca de rotinas (*framework*) para o *Common LISP*. Na próxima seção, abordaremos mais detalhes desta ferramenta.

4.3.1 LOOM

O LOOM oferece uma linguagem de programação de alto nível e um meio para construção de sistemas baseado em conhecimento (SBC) [MAC99]. Em 1986, O Instituto de Ciência da Informação (ISI), da Universidade da Califórnia do Sul, começou o desenvolvimento desta ferramenta para o desenvolvimento de SBC.

Seu desenvolvimento, foi acompanhado por diversas comunidades acadêmicas. Isto fez com que o LOOM oferecesse diversos benefícios (funções) que outros sistemas para representação do conhecimento não oferecem, como por exemplo, a possibilidade de usar diferentes tipos de raciocínio.

Um típico programa LOOM, consiste de um modelo explícito e detalhado do domínio da aplicação, junto a uma especificação estrutural da mesma, que define o

comportamento do programa. Sua linguagem de modelagem possui um rico conjunto de conceitos primitivos para a construção da representação declarativa do modelo.

Isto faz com que uma alta porcentagem das aplicações do LOOM, se torne fácil de ser expressa declarativamente. Assim as aplicações tornam-se mais fáceis de serem estendidas, modificadas e depuradas. Um fator crucial para o sucesso desta ferramenta, é sua habilidade para raciocinar efetivamente com a estrutura de representação declarativa do conhecimento.

A arquitetura de raciocínio do LOOM centra-se em uma máquina de inferência chamada "**classificador**", que tem sido um meio para [MAC99]:

1. Incrementar o poder de dedução do sistema;
2. Proporcionar uma melhora significativa no desempenho do sistema; e
3. Coordenar o uso de múltiplos pensamentos, com uma única arquitetura de software.

O trabalho do classificador do LOOM é computar o tipo de relacionamento (super-conjuntos/ subconjuntos) entre os conceitos (objetos), que são relações unárias, e computar a instanciação do relacionamento entre os conceitos e suas instâncias.

Na Figura 4.3 temos um exemplo simples desta tarefa. Suponha uma base de conhecimento, que inclua as seguintes definições e afirmações (escritas em LOOM):

```
// define um conceito chamado 'Person' (Pessoa)
(defconcept Person)
(defconcept Male)
// define o conceito 'Person-with-sons', que é derivado de 'Person' e deve possuir pelo
menos um filho do sexo masculino
(defconcept Person-with-sons
        :is (:and Person  (:at-least 1 has-child Male)))
(defconcept Person-with-two-sons
        :is (:and Person  (:exactly 2 has-child Male)))
// define a relação 'has-child'
(defrelation has-child :domain Person :range Person)
```

```
// instancia os conceitos e relações
(tellm (Person Fred))
(tellm (has-child Fred Sandy))
(tellm (Male Sandy))
```

Figura 4.3 – Exemplo Loom

Na Figura 4.3, o classificador irá determinar que **Person-with-two-sons** é um 'subconceito' de **Person-with-sons**. Ele também determinará que **Fred** é uma **instancia** do conceito **Person-with-sons**.

Para uma melhor compreensão do estudo de caso, apresentado nos próximos capítulos, vamos na próxima sessão apresentar alguns comandos da linguagem LOOM, ressaltando que como o Loom é uma biblioteca de rotinas do *Common Lisp*, é possível utilizar os comandos *Common Lisp* em qualquer parte do programa.

4.3.1.1 Alguns Comandos LOOM

Vamos a seguir, apresentar alguns comandos da linguagem LOOM, para que haja uma maior compreensão dos próximos capítulos.

- **defconcept**: define um conceito do mundo real;
- **defrelation**: define a relação entre os atributos de um conceito e o próprio conceito, isto é, o intervalo de valores que as instâncias deste atributo podem encontrar-se;
- **defset**: define um conjunto de valores;
- **defmethod**: define um método, que possui um título obrigatoriamente (:title), uma situação para a execução do método (:situation), que não é obrigatório e o corpo do método (:response);
- **tellm** (ou **tell**): atualiza a base de conhecimento com alguma informação;
- **create**: cria uma nova instância de um conceito já existente;

- **format**: comando para imprimir em tela uma mensagem
- **read**: comando para ler um valor via teclado e simultaneamente instanciar uma variável com este valor;
- **case**: comando de desvio condicional;
- **perform**: comando para executar um método;

Capítulo 5 - Estudo de Caso: ACOMED

5.1 Introdução

Neste capítulo iremos abordar o estudo de caso realizado. Trata-se de um agente cognitivo cuja função é acessar uma base de dados a respeito de alunos, professores, cursos e disciplinas de uma determinada universidade.

A idéia de construir este agente surgiu devido às dificuldades no acesso a bases, encontradas pelo grupo de pesquisa da UFPR voltado para agentes, ao desenvolver o SISMAT [MAT00] e o AGENTE-M [GUI00].

Assim, buscamos desenvolver um agente mediador, denominado de ACOMED (Agente Cognitivo Mediador), onde o principal enfoque foi o agente em si, tornando explícitos seus aspectos de comportamento e raciocínio.

O protótipo do ACOMED está fundamentado na arquitetura BDI proposta pelo DESIRE, e que foi adaptada para o AGENTE-M [GUI00]. A implementação deste foi feita através da linguagem de representação do conhecimento LOOM.

O agente está especificado em componentes internos que interagem quando necessário, de acordo com a situação em que se encontra o agente. Desta forma, através do ACOMED, conseguimos avaliar o desempenho da arquitetura BDI adotada, identificar as facilidades e dificuldades de implementação, as vantagens e desvantagens de sua utilização, além das diferenças de desempenho em relação ao AGENTE-M [GUI00].

No desenvolvimento deste estudo de caso existem algumas diferenças na filosofia do AGENTE-M em relação aos estados mentais do nosso agente mediador, como a determinação do desejo. O desejo primordial do ACOMED é sempre o mesmo: localizar uma informação e retorna-la ao agente solicitante, enquanto que no AGENTE-M o desejo varia conforme o estado em que o agente se encontra. Porém, o ACOMED pode desejar a alteração da base de dados também. Isto faz com que os compromissos assumidos pelo ACOMED se modifiquem. Essa mudança depende do agente que solicitou a informação.

Neste trabalho também damos uma ênfase maior nos componentes gerenciamento de interação com outros agentes e gerenciamento de interação com o mundo.

5.2 Visão Geral do ACOMED

A função do ACOMED é, após receber o pedido de localização de informação (através de uma ontologia específica), verificar se é possível encontrar a mesma. Em caso afirmativo, ele procede para a busca e passa os resultados encontrados ao agente solicitante.

O processo de pesquisa para encontrar uma informação envolve conhecimento sobre a base de dados. Assim, o agente mediador deve ter esse conhecimento e saber manipulá-lo para simular as capacidades que um humano apresenta, como a crença na existência de um aluno e o desejo de localizar seus dados para consultá-los.

Para realizar a pesquisa de uma informação é proposto um ambiente, como o mostrado na Figura 5.1, composto basicamente por três elementos:

- Aplicações do usuário que desejam localizar uma informação. No nosso estudo de caso, esta aplicação é composta de três agentes: o *login*, o *consulta* e o *alteração*.
- Um agente mediador que deseja localizar a informação solicitada pelos agentes do ambiente. Para isso, ele analisa a existência de tal informação e compromete, se possível, a transmitir as informações encontradas.
- Uma base de dados com informações sobre alunos, disciplinas, cursos e professores.

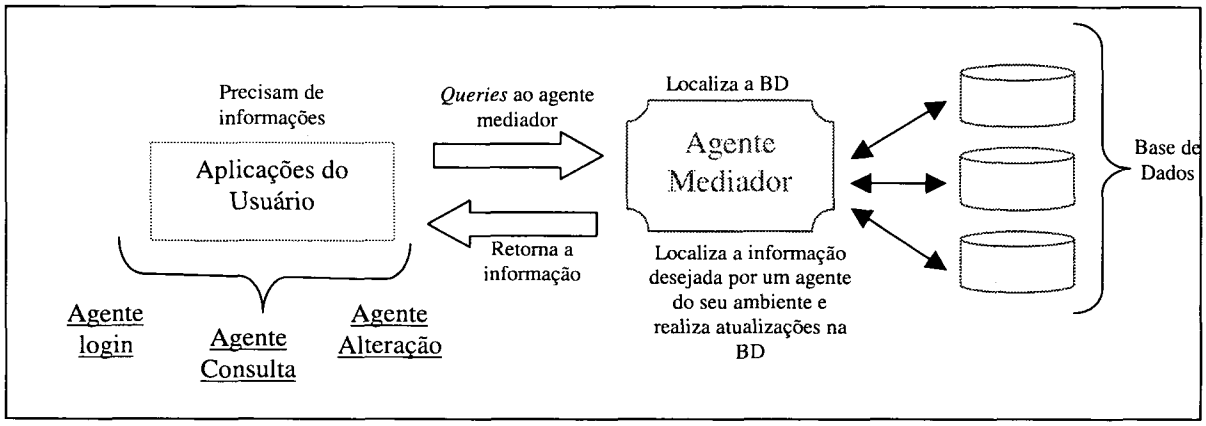

Figura 5.1 – Visão Geral do Estudo de Caso

As aplicações do usuário precisam das informações que estão armazenadas nas bases de dados. Tal aplicação é composta de três agentes:

1. O agente *login* que deve verificar se as informações que o usuário digitou ao entrar no sistema estão consistentes. Além disso, ele precisa saber se este usuário tem direito de alterar a base de dados ou não e comunicar isto para ele (usuário).
2. O agente consulta que tem como meta localizar uma informação do mundo e verificar os dados armazenados sobre esta informação.
3. O agente alteração que tem como objetivo, verificar se uma informação existe e conseqüentemente, modificar os dados armazenados sobre esta informação.

Assim, ao desejar a localização de uma determinada informação, o agente deve informar o código da mesma ao agente mediador que, raciocinando em cima de seus conhecimentos e da base de dados, valida a informação e transmite os resultados ao agente solicitador.

Por exemplo, se o agente usuário deseja verificar as informações de um determinado aluno, ele deve programar o 'agente consulta' para esta tarefa. Este por sua vez, chama o ACOMED para auxiliá-lo nesta função, informa a ele o código do aluno desejado e o ACOMED encarrega-se de verificar se a informação existe ou não. Caso ela exista, ele deve enviar as informações encontradas ao agente que as solicitou. Neste caso, o agente consulta.

Nosso objetivo é projetar o agente mediador, que denominamos ACOMED, através de uma arquitetura BDI, representando explicitamente o seu relacionamento com o mundo (base de dados) e com os demais agentes do ambiente. Na seção seguinte apresentamos o projeto do ACOMED.

5.3 Projeto do ACOMED

Uma das principais características de um agente inteligente é sua capacidade de raciocínio, controle de suas tarefas e de gerenciamento da interação com outros agentes e com o mundo. Assim, estruturamos o ACOMED em 4 componentes internos, já que o DESIRE permite a generalização do seu modelo: Controle do Próprio Processo, Gerenciamento do Mundo (onde agrupamos o módulo Manutenção das Informações do Mundo), Gerenciamento de Agentes (onde agrupamos o módulo Manutenção das Informações dos Agentes) e Tarefas Específicas do Agente.

Cada componente possui em sua estrutura informações de entrada e de saída e uma base de conhecimento. As informações de entrada correspondem às informações que são transferidas para os componentes e as de saída, são as informações exportadas pelo mesmo. As bases de conhecimento contêm o meta-conhecimento que define o raciocínio do componente, determinando qual o próximo componente que será ativado (próximo_componente) e qual informação (próximo_link) será transferida a ele.

Vamos agora discutir a estrutura e a função de cada componente do ACOMED.

5.3.1 Componente Controle do Próprio Processo:

Este é o principal componente dentro do agente, uma vez que define suas características e determina o enfoque para o seu comportamento, com base nas especificações dos seus estados mentais.

Considerando que o ACOMED foi modelado através de uma arquitetura BDI foram estabelecidos os seguintes subcomponentes: determinação de crenças, de desejos e de intenções (compromissos). Assim, cada estado mental é definido separadamente e o relacionamento entre eles ocorre através da interação desses subcomponentes.

Dentro do componente 'determinação de intenções e compromissos' definimos um subcomponente responsável pelos planos que o agente deve estabelecer para realizar seus desejos. O diagrama a seguir (Figura 5.2) representa o refinamento do componente

Controle do Próprio Processo do ACOMED, baseado em [BRA98a, BRA98b, BRA99a, BRA99b] :

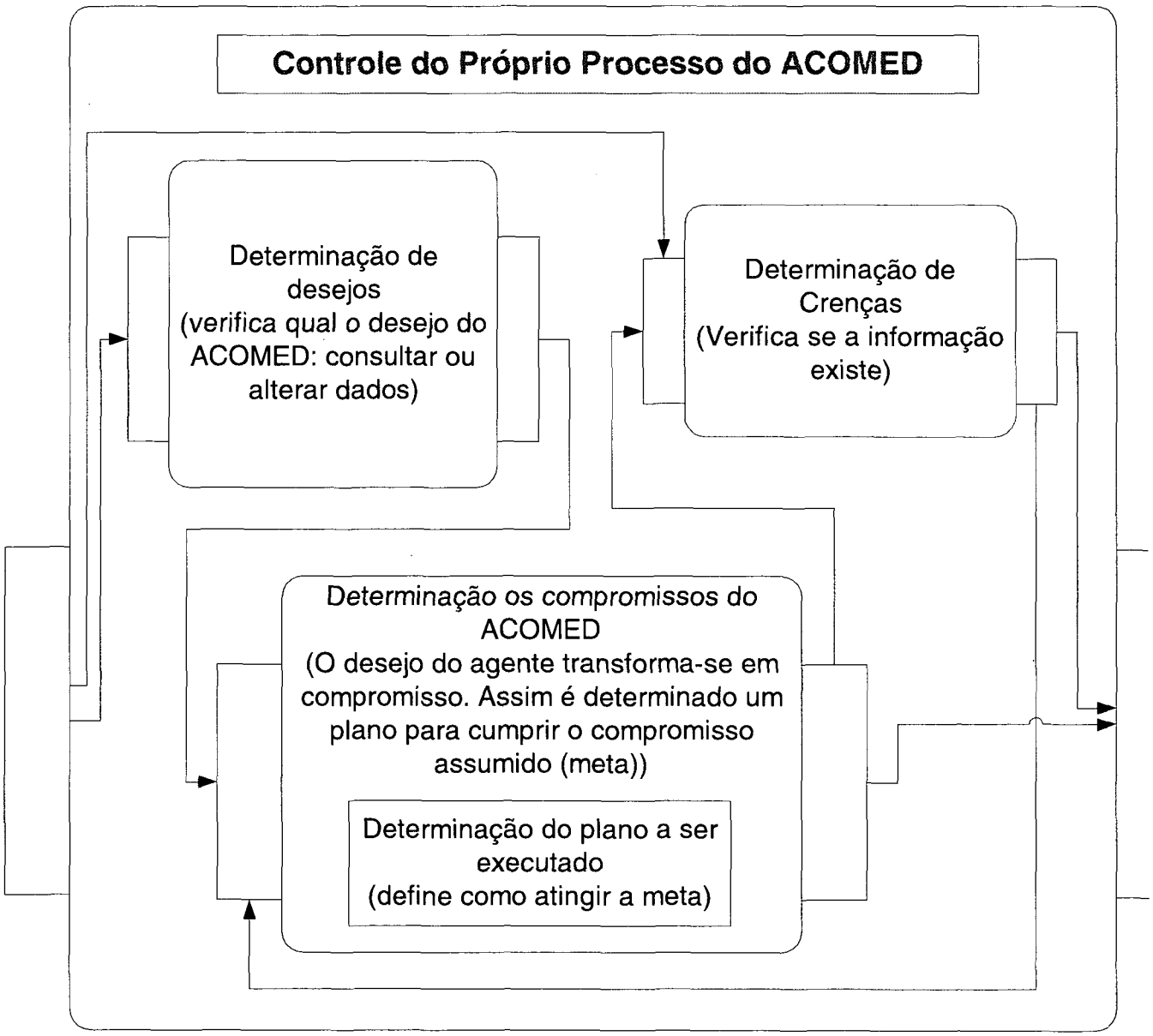

Figura 5.2 – Refinamento do Controle do Próprio Processo do ACOMED

5.3.1.1 Determinação das Crenças do ACOMED

Para a tarefa de determinação das crenças precisamos de um meta-raciocínio explícito. O conhecimento do ACOMED para esse propósito está baseado nas seguintes fontes:

- **Crenças internas**: são as crenças que o ACOMED possui inerentemente, ou seja, são os fatos iniciais. Na Figura 5.3 apresentamos o meta-conhecimento que estabelece a crença de que existe uma base de dados, onde estão armazenadas as informações.

```
Se crença-interna (X: existe BD)
        então crê (X: existe BD)
```

Figura 5.3 - Crença interna do ACOMED

Os parâmetros passados para este módulo crença-interna na Figura 5.3 são X, referente ao agente que possui a crença interna e existe BD, que é no que o agente X crê.

- **Crenças baseadas em comunicação**: são as crenças que o ACOMED possui através da comunicação com os demais agentes. Considerando que o fato comunicado por um agente é verdadeiro, gera-se uma crença neste fato.

O meta-conhecimento apresentado na próxima figura estabelece que o fato comunicado pelo agente é verdadeiro, o que leva o ACOMED a acreditar nele. Os parâmetros passados para o módulo fato_comunicado_por_agente são X, que é o agente que comunica algum fato; Y, que significa o agente que recebe a comunicação e Z, que é o fato propriamente dito.

Se fato_comunicado_por_agente (X, Y : Z)

então crê (X: agente_deseja_localizar_dados)

onde Z: código da informação desejada,
ou
dados para login

Figura 5.4 – Crença do ACOMED baseado no fato comunicado por outro agente

- **Crenças baseada na observação do mundo:** são as crenças obtidas através da verificação de um fato no mundo. Esta verificação é feita pelo componente gerenciamento de interação com o mundo e é comunicada ao processo de determinação de crenças, que passa a crer ou não no fato, conforme o resultado obtido pelo componente de gerenciamento do mundo.

O meta conhecimento especificado na Figura 5.5 estabelece a crença de que existem dados armazenados sobre determinado aluno, professor, curso ou disciplina. Os parâmetros passados para o módulo fato_verificado_mundo são X, referente ao agente que está verificando o fato no mundo e existência_de_dados_na_BD que é o fato que está sendo verificado.

Se fato_verificado_mundo (X: existência_de_dados_na_BD)

então crê (X: existência_de_dados_na_BD)

Figura 5.5 – Crença do ACOMED baseada na observação do mundo

Assim conseguimos estabelecer as crenças para o ACOMED utilizando as diferentes fontes de informação mencionadas anteriormente. As crenças internas estão definidas dentro da base de conhecimento do componente controle do próprio processo e as

crenças baseadas em comunicação são definidas em tempo de execução do sistema e comunicadas ao referido componente.

Em suma, as informações de entrada do subcomponente determinação de crenças são as informações comunicadas pelos demais agentes e as observadas no mundo. A saída corresponde à transferência das crenças para o componente determinação dos desejos.

5.3.1.2 Determinação dos Desejos do ACOMED

Os desejos de um agente referem-se aos acontecimentos pretendidos. Assim, o desejo primordial do ACOMED é atender a solicitação de informação feita por um agente do ambiente. Além deste desejo, o ACOMED pode também pode desejar a alteração dos dados contidos em tal informação. Tais desejos são transferidos para o componente determinação de compromissos, que traçará o plano para realizá-lo.

O meta-conhecimento especificado na Figura 5.6 expressa que se foi realizado uma solicitação de informação (seja pelo agente alteração, consulta ou *login*), o ACOMED passa a ter como desejo a localização da informação e a transmissão dos dados encontrados ao agente solicitante. O próximo passo (próximo_*link*) é transferir o desejo para o componente determinação de compromissos, que verificará as crenças do ACOMED e estabelecerá os planos para atingir essa meta, caso a crença em uma determinada informação seja verdadeira.

```
Se solicitação_informação
        então desejo (localizar_info
                        e transferir_info_para_agente_solicitante)
        e próximo_link (transfere_desejo_para_dc)
```

Figura 5.6 – Determinação de desejo do ACOMED

A entrada para esse componente são as informações vindas dos demais agentes: código da informação e intenção do agente solicitante (*login*, consulta ou alteração) (info_transferida e intenção_agente_solicitante). O átomo de saída, nada mais é que a transferência do desejo para o componente determinação de intenções e compromissos, já que o desejo leva ao comprometimento de realizar uma ação (transfere_desejo_para_dc).

A Figura 5.7 apresenta a estrutura interna do componente determinação de desejos e a especificação do desejo do ACOMED de transferir as informações armazenadas ou alterá-las, caso a avaliação da existência da informação na base de dados seja bem sucedida. Esta figura é uma generalização da Figura 5.6, pois mostra os dois desejos possíveis do ACOMED.

Observa-se ainda na Figura 5.7 que o desejo é determinado conforme a intenção do agente mediador: se a intenção dele é consultar, então o seu desejo será buscar os dados na base e transferi-lo para o agente solicitante. Se a intenção é alterar, então o desejo será de buscar a informação na base, altera-la e transferir a informação alterada para o agente solicitante. Após a determinar o desejo, ACOMED deve informar este desejo (próximo_link) para o componente determinação de compromisso.

Componente Determinação dos Desejos

Entrada: info_transferida e intenção_agente_solicitante

Saída: transfere_desejo_para_dc

Base de conhecimento desejos:

Se solicitação_informação e intenção_agente_solicitante(consultar)
então (transferir_dados_para_agente_consulta)
e próximo_link (transfere_desejo_para_dc)

Se solicitação_informação e intenção_agente_solicitante (alterar)
então (alterar_dados e transferir_dados_alterados_para_agente_alteração)
e próximo_link (transfere_desejo_para_dc)

Onde dc = determinação de compromissos

Figura 5.7 – Determinação dos desejos do ACOMED

5.3.1.3 Determinação dos Compromissos do ACOMED

O componente determinação dos compromissos do ACOMED existe para definir a estratégia para a realização de um determinado desejo do agente e está baseado em metas e planos. Assim, quando o agente possui um desejo e verifica que é possível realizar este desejo, ele estabelece um plano em função de uma crença ou uma meta já atingida.

Na ativação do plano são definidos qual o próximo conjunto alvo, que pode ser um subcomponente ou outro componente e também qual link de informação será ativado. Na Figura 5.8 é mostrado o meta-conhecimento que estabelece o compromisso baseado no desejo e crença do agente. Se o agente deseja localizar e transferir os dados de um aluno, sua meta será localizar e transferir tais dados. Para tanto, verifica-se a crença nestes dados e caso a mesma seja verdadeira, estabelece-se um plano que requer a ativação do componente gerenciamento do mundo para a localização dos dados. Uma vez que eles são encontrados, assume-se o compromisso de transferí-los, sendo necessário então, ativar o componente tarefas especificas. Este por sua vez, irá ativar o gerenciamento de agentes para transferir os dados.

```
Se desejo (X : transferir info)
        então meta_pretendida (X: transferir info)
            e próximo_componente (gerenciamento_do_mundo, ativar)
            e próximo_link (transfere_info)
            e se crê(info, verdadeira)
                então compromisso(X: transferir info, A:usuário)
                    e plano_pretendido(X:mostrar info, A:usuário)
                    e próximo_componente (tarefas_específicas, ativar)
                    e próximo_link (transfere_compromisso)

onde info = informação requerida pelo usuário
```

Figura 5.8 – Determinação do compromisso, baseado em um desejo do agente

Na Figura 5.9 temos a expressão que determina um compromisso baseado em uma meta atingida. Esse meta-conhecimento indica que se a meta que havia sido estabelecida, de alterar dados, foi atingida, então o agente passa a ter um novo desejo: mostrar o resultado da alteração. Assim, ele assume um novo compromisso com o usuário, mostrar os dados alterados; estabelece um novo plano para mostrar os dados alterados, e determina qual será o próximo componente a ativar (tarefas específicas).

```
Se meta_atingida (X: alterar_dados)
        então  novo_desejo (X: mostrar_dados_alterados)
               e novo_compromisso (X: mostrar_dados_alterados)
               e plano_pretendido (X: mostrar_dados_alterados)
               e próximo_link (tarefas_específicas, ativar)
```

Figura 5.9 – Especificação do compromisso baseado em uma meta atingida

5.3.2 Componente Gerenciamento do Mundo

No Componente Gerenciamento do Mundo modelamos a interação do ACOMED com o mundo (o mundo base de dados), realizando observações e atualizando o seu estado em função dos fatos informados pelos demais agentes do ambiente.

Esse componente define qual observação deve ser feita e qual o tipo de informação que estamos procurando: informação sobre aluno, professor, curso ou disciplina. Assim, a informação de entrada é o código da informação que procuramos (código_info) e o átomo gerado para saída são as informações geradas para o mundo (info_gerada_para_mundo).

O meta-conhecimento especificado na Figura 5.10 estabelece que se o fato informado ao componente indicar que ele deve localizar dados cadastrados, então ele deve importar do mundo tais dados, procurando-os na base de dados.

A Figura 5.11 mostra que quando o componente gerenciamento do mundo é ativado, recebendo a informação de que deve atualizar a base, então deverá proceder-se a alteração do estado do mundo, ajustando a base em questão para a nova situação.

Podemos observar na Figura 5.11 que se o fato informado ao ACOMED for atualizar base (X: atualizar_base), ele deve importar do componente gerenciamento de agentes, qual informação ele deve alterar; em seguida, ele deve atualizar o mundo.

```
Se fato_informado (X: localizar_info)
        então  verifica_mundo(X: categoria_info, localização_info)
               e importar_mundo (X:dados_cadastrais,  BD: BD_localizada)
               e próximo_link (importa_informação_do_mundo, X)
```

Figura 5.10 – Especificação da interação do ACOMED com o mundo

Componente manutenção de informação do mundo

Entrada: info_gerada_pelo_agente_X
Saída: info_gerada_para_mundo

Base de conhecimento manutenção de informação do mundo:

```
Se fato_informado (X : atualizar_base)
        então importar (gerenciamento_de_agentes, dado_para_alterar)
               e alterar_mundo (X: dado_verificado, BD)
```

Figura 5.11 – Meta-conhecimento do Componente Gerenciamento do Mundo

5.3.3 – Componente Gerenciamento de Agentes

O componente gerenciamento de agentes mantém informações sobre a interação do ACOMED com os demais agentes do ambiente. As ações que esse componente deve realizar dependem do fato que lhe é comunicado, que pode ser originado pelo próprio agente ou pelo agente usuário.

O fato comunicado pelos agentes do sistema pode ser: sua identificação (nome e senha), o código da informação que deve ser consultada ou alterada, seu desejo (alterar ou consultar uma informação) e no caso de alteração, qual dado alterar e a nova informação a ser armazenada.

Já o ACOMED transfere informações para o agente solicitante durante todo o processo de localização de uma informação, mostrando sempre os resultados de sua observação e raciocínio, mantendo o agente solicitante a par de como está se sucedendo o processo de localização de uma informação. Esses resultados informados pelo agente mediador, que correspondem aos átomos de saída, são a validade das informações de *login* (crença nesta informação), o tipo de acesso que o usuário possui: somente consulta ou consulta e alteração, a validade do código da informação desejada (crença no código), o seu desejo de atender a solicitação de busca de informação ou atualização do mundo, o compromisso assumido com o agente solicitante; em caso de alteração: qual dado o 'agente alteração' deseja alterar e o pedido do novo dado.

O meta-conhecimento mostrado na Figura 5.12 especifica que se o fato comunicado for proveniente do próprio ACOMED (Agente:X) e tiver como conteúdo transferir_dados_armazenados, o agente mediador deverá exportar esses dados para o agente solicitante (Agente:Z).

```
Se fato_comunicado_por (X:transferir_dados_armazenados, Agente: X)
        então  exportar(X:dados_armazenados, Agente: Z)
```

Figura 5.12 – Especificação de interação do ACOMED com o agente usuário

A Figura 5.13 apresenta a estrutura do componente gerenciamento de agentes e destaca o meta-conhecimento que determina que, se o fato comunicado ao ACOMED (Agente:A) for alteração de dados, sua ação será a de importar do Agente:Z (neste caso, agente alteração) o item a ser alterado.

Componente Gerenciamento de Agentes

Entrada: fato_comunicado
Saída: info_comunicada_agente

Base de conhecimento gerenciamento de agentes

Se fato_comunicado_por (X:alterar_dados, Agente: A)
então importar (X:dado_para_alterar, Agente: Z)

Figura 5.13 – Meta-conhecimento do componente gerenciamento de Agentes

5.3.4 – Componente Tarefas Específicas do Agente

Neste componente foram modeladas as tarefas próprias do domínio do agente, que são definidas em subcomponentes específicos, existindo também um subcomponente que avalia o estado do processo.

O diagrama da Figura 5.14 apresenta a decomposição do componente tarefas específicas do agente. O conhecimento do controle da tarefa desse componente determina que inicialmente o componente 'avalia o estado do processo' seja ativado, para identificar qual 'tarefa específica' deverá ser realizada no momento, de acordo com a informação recebida durante sua ativação.

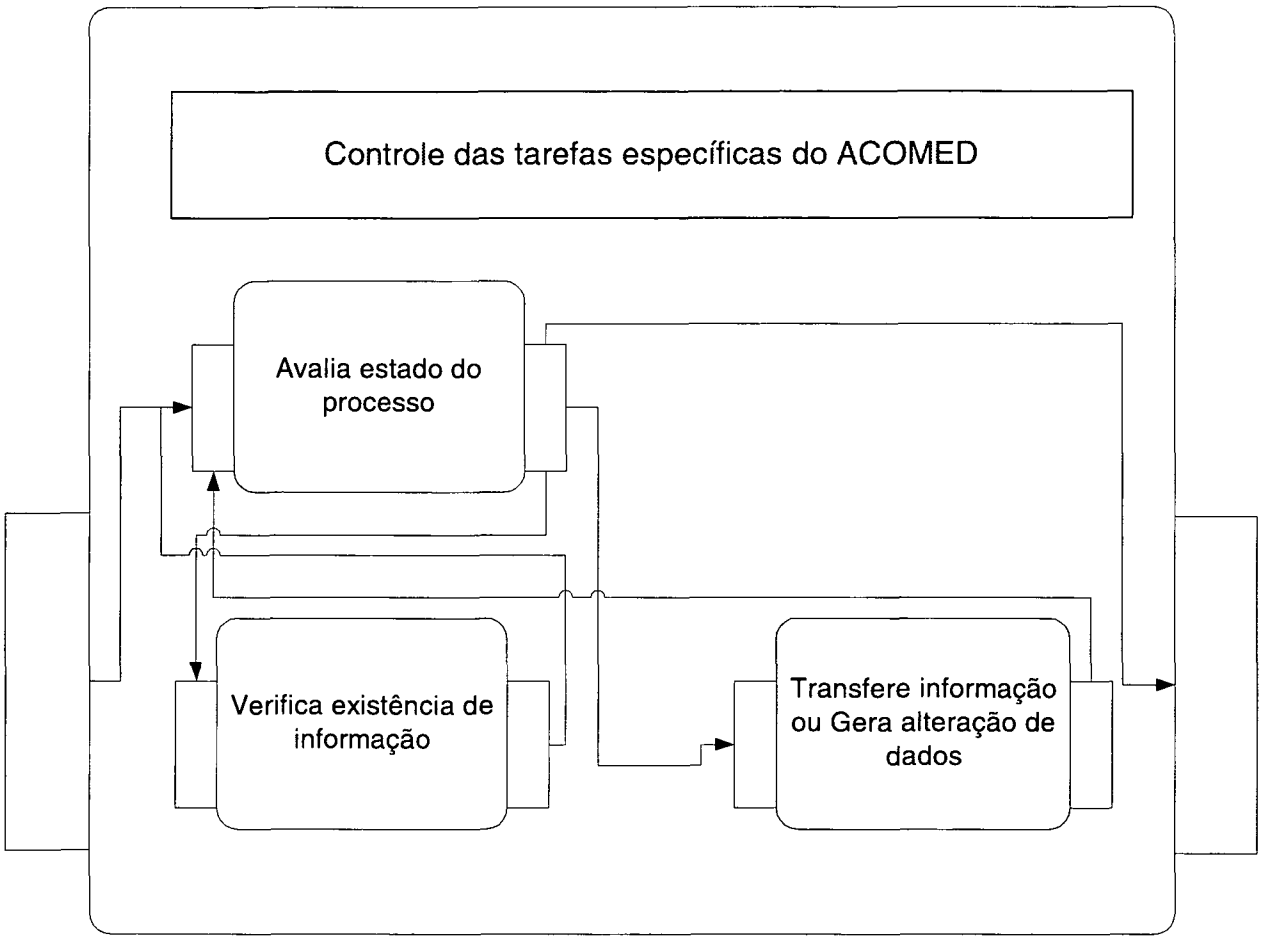

Figura 5.14 – Tarefas Específicas do ACOMED

A Figura 5.15 mostra a decomposição do subcomponente 'transfere informação ou gera alteração de dados' em três outros subcomponentes. O conhecimento desse componente determina que inicialmente se avalie o processo a ser executado. Essa avaliação é feita perante a informação recebida na sua ativação. Os subcomponentes 'transfere informações' e 'gera alteração' são ainda decompostos em outros subcomponentes para que seja feita uma avaliação de qual o tipo da informação a transferir ou alterar: aluno, professor, curso ou disciplina.

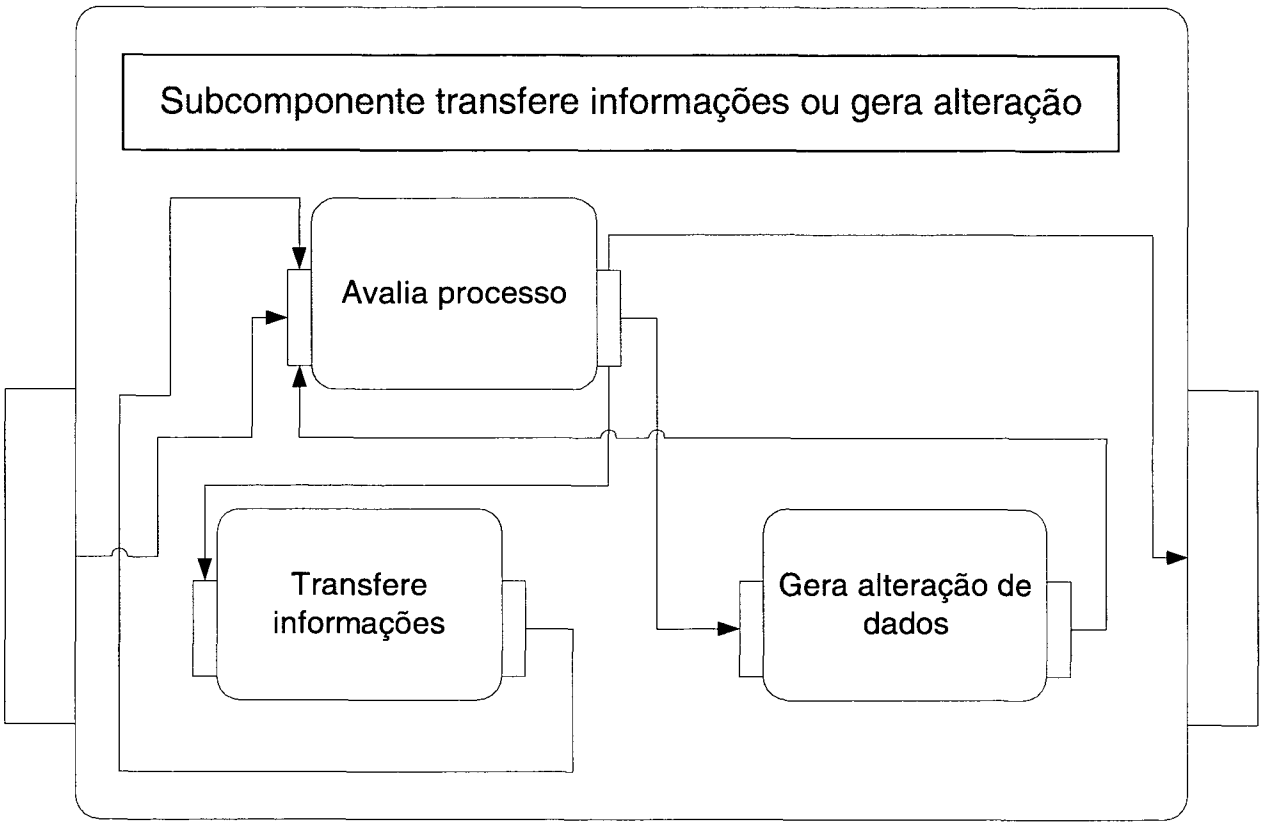

Figura 5.15 - Decomposição do subcomponente transferir informações ou gerar alteração

A Figura 5.16 mostra o meta-conhecimento para o subcomponente gera alteração de dados, no caso de se tratar de uma alteração dos dados de um professor. Vemos que se a tarefa do ACOMED for alterar informações de um professor, então o componente gerenciamento de agentes deve ser ativado para pedir ao agente solicitante (Agente:Z) qual item deve ser alterado e qual o novo conteúdo deste item (novo_dado). Após isto, ele deve ativar o componente gerenciamento do mundo para efetivar a atualização. Em seguida, o ACOMED deve transferir (próximo_link) o resultado da alteração para o agente solicitante (Agente:Z).

A tarefa de gerar uma alteração é realizada em função da avaliação da informação que se quer alterar e depois em cima do dado que se deseja modificar. Por exemplo, o agente alteração transfere o código de um professor para alteração. O ACOMED deve raciocinar em cima deste código e concluir que o mesmo pertence a um professor. Com esta informação ele continua seu raciocínio e verifica quais os dados do professor o agente alteração pode modificar (nome, endereço, telefone,...). Assim, ele pede a nova informação e continua a tarefa alterar, atualizando a base de dados.

Subcomponente Gerar alteração professor

Entrada: código_professor_válido
Saída: resultado_alteração

Se tarefa_ACOMED(alterar_professor)
então gerenciamento_de_agente {[pedir(item_para_alterar, Agente: Z)
e pedir (novo_dado, Agente: Z)], ativar}
e gerenciamento_do_mundo (ativar)
e próximo_link (X:transfere_resultado_alteração, Agente:Z)

Figura 5.16 – Meta Conhecimento do subcomponente gerar alteração professor

É importante salientarmos que se, para cooperar com outros agentes em determinado ambiente for atribuída uma nova tarefa específica para o ACOMED, basta criar um novo subcomponente dentro do componente tarefas específicas do agente, com as novas especificações da nova tarefa, e ajustar o componente de avaliação do estado do processo. Os demais componentes do ACOMED não precisam necessariamente ser alterados em função da nova tarefa específica.

Após descrevermos os componentes do ACOMED, apresentamos na próxima seção o fluxo de raciocínio (simplificado) do mesmo.

5.4 Fluxo de Raciocínio do ACOMED

É válido lembrar que para estabelecer-se este fluxo foi necessário determinarmos e seguirmos uma seqüência lógica de raciocínio, representada na Figura 5.17: o usuário deseja obter informações. Assim ele programa um agente para executar a tarefa que, pede

auxílio para o ACOMED. O ACOMED por sua vez, deseja atender o pedido do agente que solicitou sua ajuda. Logo, este desejo é transferido para o componente de intenção e compromisso que após verificar a crença nos dados fornecidos, traça o plano de transferência da informação.

Antes de assumir um compromisso de localização ou alteração dos dados com os agentes do ambiente, o ACOMED verifica suas crenças e caso elas sejam verdadeiras assume o referido compromisso. Então é traçado um plano para o cumprimento do acordo, que determina quais tarefas o ACOMED deve executar até atingir sua meta.

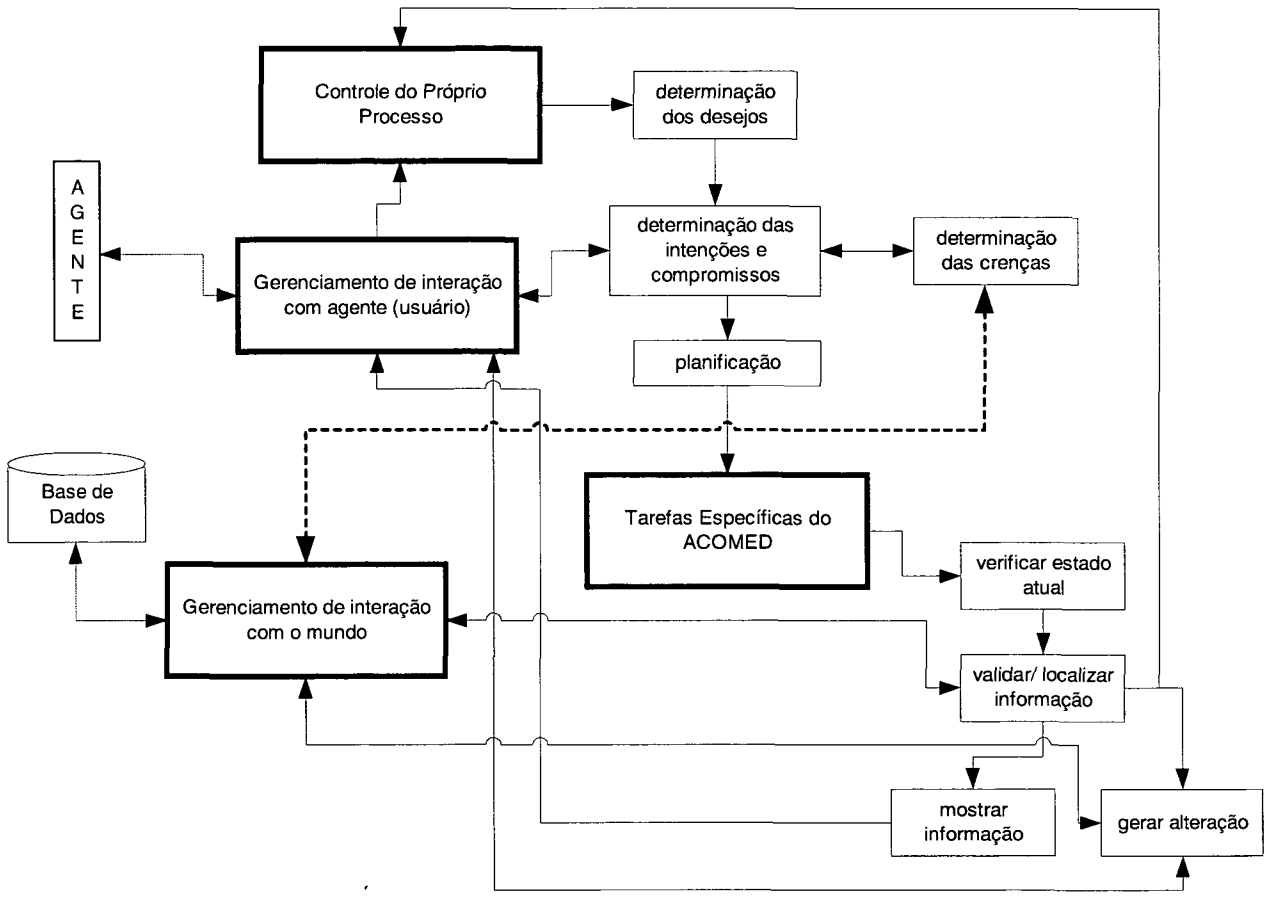

Figura 5.17 – Fluxo de raciocínio do ACOMED

Já tendo discutido os estados mentais e a filosofia de raciocínio do ACOMED, bem como a arquitetura utilizada para o seu desenvolvimento, no próximo capítulo vamos discutir os aspectos relevantes da sua implementação, como por exemplo, a implementação das crenças, dos desejos e intenções.

Capítulo 6 - Implementação do ACOMED

6.1 - Introdução

Neste capítulo vamos abordar os aspectos relevantes da implementação do ACOMED e mostrar os resultados obtidos com a mesma. Utilizamos para implementar o nosso agente a linguagem LOOM que corresponde a uma biblioteca de subrotinas LISP [LOOM91]. Por questões de limitação financeira utilizamos a versão *trial* 5.0 do Allegro Commom Lisp. O ambiente base foi um PENTIUM III 550 Mhz com 128 MB de memória e Sistema Operacional Windows NT 4.0, *service pack* 6.0.

Para a implementação do ACOMED estabeleceu-se que o comportamento do agente, definido na base de conhecimento de cada componente, fosse representado através de métodos, que correspondem a conjuntos de operações associados à determinada ação. A representação do conhecimento do agente efetuou-se através de conceitos e instâncias.

A chamada entre os métodos corresponde à interação entre os próprios componentes e entre eles, o mundo e o agente usuário. Vamos iniciar mostrando como foi feita a representação do mundo, para que possamos entender melhor alguns passos realizados pelo ACOMED, na tentativa de atingir sua meta de localizar uma informação. Em seguida, vamos ver como foram implementados os estados mentais do ACOMED.

6.2 - Implementação do Mundo

Quando nos referimos ao mundo, estamos nos referindo as bases de dados e de conhecimento existentes. Cada objeto do mundo é tratado como um conceito, o qual é compreendido perfeitamente pelo ACOMED.

Assim como no SIMS, um modelo de cada base de informação é criado para descreve-la ao sistema. Além disso, um modelo do domínio é construído para descrever objetos e ações que são significantes na performance das tarefas do domínio da aplicação. A coleção de termos do modelo do domínio forma o vocabulário usado para caracterizar o conteúdo de uma base de informação.

Assim, para termos a expressividade necessária referente ao mundo do agente, utilizamos o comando ***defconcept*** para a implementação dos referidos conceitos. Porém, sabemos que alguns conceitos precisam de outros para ser compreendido. Assim, a linguagem LOOM utiliza-se da noção de conceitos primitivos e não primitivos. A escolha de qual conceito deve ser primitivo ou não, depende da representação do objeto que o conceito fará.

Sabe-se também que os conceitos podem possuir atributos, como por exemplo, um aluno tem nome e data de nascimento. Para expressarmos esse relacionamento (conceito-atributo) utilizamos o comando ***defrelation***.

Vale mencionarmos que alguns conceitos possuem restrições. Esse fato também pode ser representado através das funções do LOOM.

Desta forma, estas definições, relacionamentos e restrições são adicionadas a base de conhecimento e são "compiladas" pelo LOOM, cujo resultado é uma taxonomia de termos específicos do domínio do agente.

Assim, conseguimos definir o conceito aluno, mostrado na Figura 6.1(a). Esta definição expressa que existe no mundo um objeto chamado aluno que é primitivo e possui certos atributos: ra.aluno, nome.aluno, curso.aluno, endereço.aluno, cidade.aluno, etc. Para cada atributo é necessário estabelecer um relacionamento com o conceito, conforme a Figura 6.1(b).

```
(defconcept aluno :is-primitive
        (:and  (exactly 1 ra.aluno)
               (exactly 1 nome.aluno)
               (exactly 1 curso.aluno)
               (exactly 1 endereco.aluno)
               (exactly 1 cidade.aluno)
               (exactly 1 data-nasc.aluno)
               (exactly 1 ano-entrada.aluno)
               (at-least 0 telefone.aluno) ) )
```

Figura 6.1(a) – Definição da crença aluno

```
(defrelation nome.aluno
        :domain aluno    :range string)
```

Figura 6.1(b) – Definição da relação nome.aluno

Sabe-se também que existem vários alunos no ambiente. Este fato é representado através da instanciação do conceito aluno, como mostrado na Figura 6.2. Nela, atualizamos a base através do comando ***tellm***, criamos uma nova instancia de aluno através do comando ***create.*** O componente gerenciamento do mundo traz a informação instanciada quando necessário.

```
(tellm  (create a1 aluno)
        (codigo.aluno a1 123456)
        (nome.aluno a1 "João da Silva")
        (curso.aluno a1 c1)
        (endereco.aluno a1 "Av. Monte Castelo, 001 ")
        (cidade.aluno a1 "Curitiba")
        (dia-nasc.aluno a1 14)
        (mes-nasc.aluno a1 'jan)
        (ano-nasc.aluno a1 1977)
        (ano-entrada.aluno a1 1999)
        (telefone.aluno a1 3396070) )
```

Figura 6.2 – Instanciação do conceito aluno

Pode-se observar o fato de que, por exemplo, o dia de nascimento do aluno (dia-nasc.aluno) deve pertencer a um intervalo de dias (1 a 31) e só possui um valor. Restrições de como essa, puderam ser representadas através do comando ***defset***, ***:characteristics*** e ***:single-valued*** como mostrado na Figura 6.3.

```
(defset day
        :is (:interval++ 1 31)
        :characteristics :single-valued)
```

Figura 6.3 – Restrição sobre a instanciação do conceito 'day'

Dessa forma, definimos o mundo do ACOMED e conseguimos criar uma ontologia para o mesmo, o que permite sua re-utilização em diversos ambientes que utilizem a mesma ontologia.

Na próxima seção vamos explicar como foram implementados as crenças do ACOMED.

6.3 Implementação das crenças do ACOMED

Em nível de implementação, o único tipo de crença que devemos codificar são as crenças observadas. Isto é, o ACOMED acredita na existência de dados de um determinado aluno, por exemplo, através da observação do mundo. Ele deve verificar o mundo e se existir a instância a respeito do aluno procurado ele passa a acreditar na existência de possíveis dados armazenados sobre este aluno. Isto foi representado em LOOM através de métodos que verificam as bases e instanciam a crença do ACOMED, conforme o resultado obtido no mundo. Um (1) significa que a informação foi encontrada e zero (0) que ela não foi encontrada. Isto está representado na Figura 6.4.

As crenças internas já são inerentes ao ACOMED. Ele sabe que existe uma base de dados e que nela existem dados sobre alunos, professores, disciplinas, etc. Então ele acredita nesta existência.

Quando falamos em crenças comunicadas, estamos nos referindo as informações vindas de outros agentes, informações estas, que o ACOMED acredita serem sempre verdadeiras.

```
        (defmethod verifica-mundo (?R)
                :title "cre em alunos"
                :situation (aluno ?R)
                :response (
                                (format t "~%*** EXISTEM INFORMACOES SOBRE
ALUNOS. ***~%~%")
                                (setq ?Crenca '1) ) )

        ;*****************************************************************
***
        (defmethod verifica-mundo (?R)
                :title "nao cre"
                :response (
                                (format t "~%*** NAO E' POSSIVEL LOCALIZAR
INFORMACAO. ***~%~%")
                                (setq ?Crenca '0) ) )
```

Figura 6.4 – Implementação da observação do mundo e comunicação da existência da instância procurada

6.4 Implementação dos desejos do ACOMED

A determinação dos desejos é um subcomponente do controle do próprio processo. Este subcomponente foi implementado através de um método com o mesmo nome. Conforme o agente que faz esta solicitação, é determinado o desejo do agente: se a solicitação partiu do agente *consulta* ou *login*, o desejo do ACOMED é localizar a informação e retorná-la ao agente consulta ou login, respectivamente; se a solicitação partiu do agente *alterar*, o desejo do ACOMED é localizar a informação, alterá-la e retorná-la ao agente alterar.

Assim, o ACOMED verifica sua crença para ver se é possível realizar seu desejo. Isto é, ele verifica no mundo (através do componente gerenciamento do mundo, discutido na seção 6.5) se a informação que ele deseja validar existe. Caso a crença seja verdadeira

ele deve assumir o compromisso de que vai realizar a tarefa com o agente que o invocou. Para isto é chamado o componente determinação de compromissos.

Se a não houver crença na existência da informação, aborta-se o processo e o programa é finalizado.

Para determinação da crença é válido lembrarmos que o agente deve raciocinar em cima do código da informação para determinar a qual categoria ela pertence (login, aluno, professor, etc.).

Na Figura 6.5 mostramos como foi implementado o componente desejo. Ele é o primeiro a ser chamado, quando o ACOMED é invocado para realizar alguma tarefa, pois sabe-se que um agente mediador tem como meta localizar uma informação no mundo. Este é um desejo inerente ao mediador, que não se modifica, por isso esse método é o primeiro a ser chamado. Porém, só é possível atingir uma meta se tiver crença de que isso é possível. Por esta razão a determinação da crença é feita antes de se assumir um compromisso.

O método 'desejo', mostrado na Figura 6.5, tem como parâmetro de entrada o código da informação solicitada (?R). Conforme o agente que fez a solicitação de informação (case ?Agente) é determinado o desejo do ACOMED (setq ?Desejo_Acomed). Após isto é disparado o método verifica-mundo (perform (verifica-mundo ?R)) para verificar a existência da informação na base de dados e determinar a crença nisto. Conforme a crença determinada (case ?Crença) assume-se o compromisso com o agente de que a informação solicitada será buscada ou finaliza-se o sistema.

```
(defmethod desejo (?R)
        :title "determina desejo ACOMED"
        :response ( (case ?Agente
                        ('login
                                (format t "~%*** Desejo ACOMED: Atender soliticitacao
de  verificação de login ***~%")
                                 (setq ?Desejo_Acomed  'localizar_e_mostrar_info))
                        ('localizar
```

```
                    (format t "~%*** Desejo ACOMED: Atender soliticitacao
de consulta a informacao ***~%")
                    (setq ?Desejo_Acomed 'localizar_e_mostrar_info))
                ('alterar
                    (format t "~%*** Desejo ACOMED: Atender solicitacao de
alteracao da BD ***~%")
                    (setq ?Desejo_Acomed 'localizar_alterar_e_mostrar_info))
                )
                (perform (verifica-mundo ?R))
                (case ?Crenca
                    ('1  (perform (assume-compromisso ?R) ) )
                    ('0  (perform (finalizar) ) )
                ) )
```

Figura 6.5 – Determinação do desejo

6.5 Implementação dos compromissos

Do mesmo modo que os demais componentes, a implementação dos compromissos foi feita através de um método. O compromisso é assumido caso seja possível realizar o desejo do agente. Esta idéia está inserida na Figura 5.6 que apresenta o método "assume-compromisso".

```
(defmethod assume-compromisso (?R)
        :title "ACOMED assume compromisso"
        :response ( (format t "~%*** AGUARDE. AGENTE ATENDENDO
SOLICITACAO. ***~%~%")
                    (case ?Desejo_Acomed
                        ('localizar_e_mostrar_info
```

```
                                    (format t "~%*** ACOMED ira localizar e
retornar informacao. ***~%~%"))
                                ('localizar_alterar_e_mostrar_info
                                    (format t "~%*** ACOMED ira localizar, alterar
e retornar informacao. ***~%~%")))
                            (perform (planificacao ?R))))
```

Figura 6.6 – Determinação do compromisso

Após assumir o compromisso de que vai atender a solicitação, é necessário traçar um plano para atingir a meta pretendida. Desta forma, é disparado o método 'planificação' que determina quais tarefas o agente deve realizar até chegar na sua meta. Estas tarefas são determinadas conforme o desejo do ACOMED.

A Figura 6.7 mostra como foi implementado o meta-conhecimento contido na planificação das tarefas do agente mediador.

```
(defmethod planificacao (?R)
        :title "tarefas do ACOMED"
        :response (
                (format t "~%*** ACOMED DETERMINANDO PLANO PARA
                                ATINGIR META. ***~%~%")
                (case ?Desejo_Acomed
                        ("localizar_ e_mostrar_info
                                (perform (busca-info ?R))
                                (perform (mostra-info ?R)))
                        ('localizar_alterar_e_mostrar_info
                                (perform (alterar ?R)))
                )))
```

Figura 6.7 – Definição das tarefas para atingir a meta

6.6 Implementação do Gerenciamento do Mundo

O componente gerenciamento do mundo foi implementado através de vários métodos que acessam a base de dados, como o método mostrado na Figura 5.4. Além deste método, temos por exemplo, um outro método que acessa a base para localizar os dados armazenados sobre um aluno, por exemplo. Este método está descrito na Figura 5.8.

```
(defmethod acessa-base (?R)
        :title "acessa dados do aluno"
        :situation (aluno ?R)
        :response (
                        (setq ?Cod (get-value ?R 'codigo.aluno))
                        (setq ?Nome (get-value ?R 'nome.aluno))
                        (setq ?Curso (get-value ?R 'curso.aluno))
                        (setq ?End (get-value ?R 'endereco.aluno))
                        (setq ?Cid (get-value ?R 'cidade.aluno))
                        (setq ?DNasc (get-value ?R 'dia-nasc.aluno))
                        (setq ?MNasc (get-value ?R 'mes-nasc.aluno))
                        (setq ?ANasc (get-value ?R 'ano-nasc.aluno))
                        (setq ?Ent (get-value ?R 'ano-entrada.aluno))
                        (setq ?Fone (get-value ?R 'telefone.aluno))
                ))
```

Figura 6.8 – Gerenciamento do mundo

É válido lembrarmos que o componente 'gerenciamento do mundo' conhece os conceitos e os relacionamentos existentes no mundo, por isso pode acessar diretamente a base e instanciar uma variável com a informação desejada.

6.7 Implementação do Gerenciamento de Agentes

A interação do ACOMED com os outros agentes do sistema é feita através de vários métodos e funções. Esta interação ocorre através dos comandos LOOM e através de alguns comandos em LISP puro, já o que LOOM nada mais é do que uma biblioteca de funções LISP.

A Figura 6.9 mostra a implementação do 'agente consulta' obtendo o código da informação que deve ser localizada e transmitindo esse código para o ACOMED ***[(perform (ACOMED (get-instance ?R))***].

```
(defun agente-consultar (&optional (?Esc))
        (format t "~% Por favor, digite o codigo da informacao para consulta: ")
        (setq ?R (read))
        (perform (ACOMED (get-instance ?R ) ) ) )
```

Figura 6.9 – Comunicação do código da informação a ser localizada

6.8 Implementação das tarefas específicas do ACOMED

A tarefa de um agente mediador é acessar informações em uma base de dados. O ACOMED desempenha este papel através de suas tarefas específicas. São elas: validar e obter os dados de uma informação (consulta) e modificação destes dados.

A implementação da tarefa de obtenção dos dados de um aluno está descrita na Figura 5.10. Ela foi implementada através de um método, assim como os demais componentes. Este método também possui como entrada o código da informação(?R) na qual deseja-se a obtenção dos dados.

```
(defmethod busca-info (?R)
        :title "obter dados"
        :response (     (perform acessa-base ?R) )
```

Figura 6.10 – Implementação da tarefa específica do ACOMED

Vamos na próxima seção mostrar o resultado obtido com a implementação do ACOMED e demais agentes do ambiente. Infelizmente não foi possível gerar uma 'versão executável' do mesmo devido às limitações do *Allegro Commom Lisp*. Assim, temos sempre que utilizar o ACOMED dentro do ambiente LOOM.

6.9 – Resultados da Implementação

A operação do ACOMED é bastante simples. Para ativá-lo basta instanciar uma variável com o código da informação desejada, conforme a Figura 6.11. Algumas outras variáveis necessárias para a operação do mesmo são instanciadas pelo agente solicitador da informação e pelo agente usuário durante a sua execução.

```
(format t "~% Por favor, digite o codigo da informacao a ser consultada: ")
        (setq ?R (read))
        (perform (ACOMED (get-instance ?R ) ) )
```

Figura 6.11 – Agente consulta e a chamada ao ACOMED

A partir deste ponto o agente torna-se autônomo. Ele valida a existência da informação e transmite os dados encontrados para o agente solicitante. Conforme as fases do sistema que estão sendo executadas, é apresentada uma mensagem em tela, conforme a Figura 6.12 onde mostramos a mensagem de desejo do ACOMED, quando este desejo é atender a solicitação de verificar os dados de login.

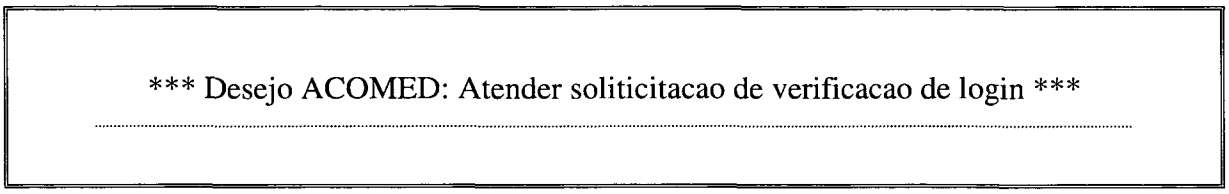

Figura 6.12 – Apresentação em tela do desejo do ACOMED

Após estabelecer o seu desejo, o ACOMED verifica a base para a validação da informação. Caso ela exista, prossegue-se o processo. Em caso contrário é gerado o fim da operação. Estas mensagens são mostradas nas Figuras 6.13 e 6.14, respectivamente.

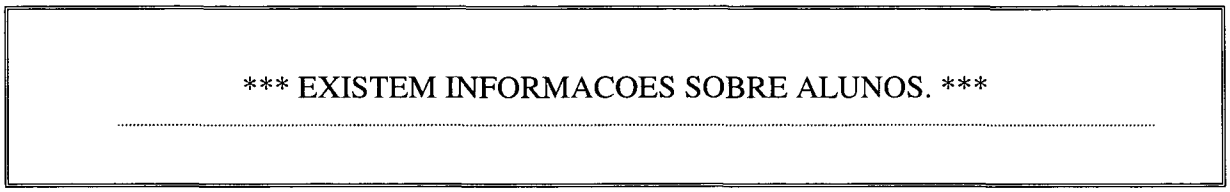

Figura 6.13 – ACOMED comunica que o código da informação existe e pertence a um aluno

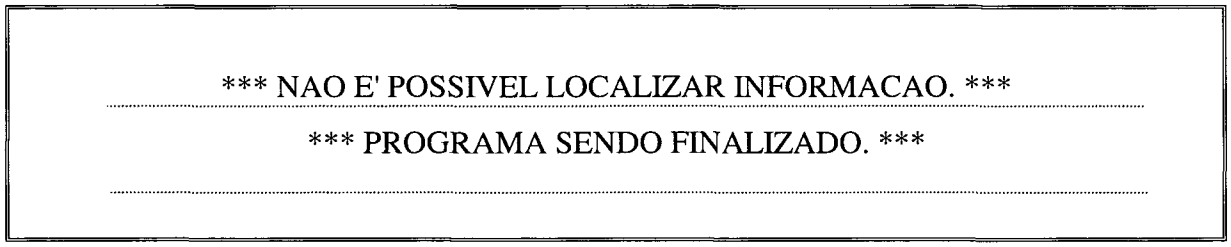

Figura 6.14 - ACOMED comunica que o código da informação não existe e que o processo está sendo abortado

Sendo a informação válida, o ACOMED assume o compromisso de atender a solicitação do agente que o invocou. A Figura 6.15 mostra a mensagem que determina este compromisso.

*** AGUARDE. AGENTE ATENDENDO SOLICITACAO. ***

Figura 6.15 – Comunicação do compromisso assumido pelo ACOMED

Após esta etapa, determina-se as tarefas que o ACOMED deve executar. Isto é comunicado em tela, com a mensagem da Figura 6.16.

*** ACOMED DETERMINANDO PLANO PARA ATINGIR META. ***

Figura 6.16 – Comunicação de que está se traçando o plano das tarefas a serem executadas

Após atingir sua meta final, o ACOMED deve transferir o resultado para o agente solicitador e este deve apresentar o mesmo em tela. A Figura 6.17 mostra o resultado da consulta de uma informação de um aluno. Após isto se encerra o sistema.

*** Informacoes armazenadas sobre aluno a23 ***

Codigo aluno.....: 23
Nome.....: João da Silva
Curso.....: Mestrado em Informatica
Endereco.....: Av. Montes Claros, 001
Cidade.....: Curitiba - Pr
Data Nascimento..: 14/01/77
Ano de Ingresso...: 1999
Fone contato.....: (41) 3396070

Figura 6.17 – Resultado da consulta aos dados de um aluno

É válido ressaltarmos que a versão da linguagem utilizada não permite a utilização de modo gráfico. Assim, todas as mensagens e resultados emitidos são apresentados em tela simples.

Tendo discutido a maneira como implementamos o nosso agente mediador e os resultados obtidos com esta implementação, resta-nos apresentar as conclusões obtidas com este trabalho, em especial com o estudo de caso. Este é o conteúdo do próximo capítulo.

Capítulo 7 - Conclusão

Este trabalho apresenta o desenvolvimento de um agente cognitivo mediador denominado ACOMED, que tem como função primordial, a localização de informações sobre alunos e professores de uma universidade. Em sua atuação, ele interage com o usuário que deseja a informação e os demais agentes do sistema (agente login, agente localização e agente alteração) e com a base de dados que contém as informações necessárias (mundo).

Este trabalho utiliza as técnicas da IAD e SMA que correspondem ao estudo da solução de problemas através da distribuição de conhecimento entre diversas entidades. Suas metas essenciais são desenvolver mecanismos e métodos que permitam agentes interagir como humanos e entender a interação entre entidades inteligentes, quer sejam elas computacionais, humanas ou ambos.

Embora o ACOMED tenha sido projetado utilizando as características do DESIRE, não foi utilizado o seu ambiente de desenvolvimento. Porém, podemos concluir que as características do DESIRE são viáveis para a utilização, mesmo não sendo desenvolvidas em seu ambiente próprio.

Ao projetarmos o ACOMED seguimos a mesma modelagem feita para o AGENTE-M, o que nos leva a concluir que o modelo proposto pelo DESIRE e adaptado para o AGENTE-M realmente pode ser reutilizado em diversas situações diferentes.

Tais projetos resultaram em uma arquitetura composta por quatro componentes principais, com raciocínio explicito sobre o próprio ACOMED, sobre o mundo e os demais agentes do ambiente.

Um modelo de cada base de informação foi criado para descreve-la ao sistema. Além disso, um modelo do domínio foi construído para descrever objetos e ações que são significantes na realização das tarefas do domínio da aplicação.

Concluímos que o projeto do agente em componentes específicos facilita a compreensão e manutenção do seu conhecimento. O uso de componentes específicos de interação com o mundo e agentes contribuiu para tornar explicito o comportamento e o raciocínio do ACOMED.

Quanto à manutenção do conhecimento, podemos verificar que a facilidade se encontra em termos que alterar apenas o componente gerenciamento do mundo, caso haja mudança na base de dados.

Uma outra facilidade da utilização de componentes é em relação às tarefas do agente. Caso determine que o agente deva realizar uma tarefa diferente das já realizadas é necessário apenas implementa-la e modificar o subcomponente 'avalia estado' pertencente ao componente 'determinação das tarefas' para que o agente passe a realizar a tarefa desejada.

A maior dificuldade encontrada na implementação do ACOMED foi a associação dos estados mentais ao comportamento do agente. Para contornarmos esta dificuldade, adotamos uma seqüência lógica de raciocínio considerando que desejamos algo (meta) com base nas crenças. Esses desejos geram compromissos que levam a ações a serem executadas para atingir-se a meta pretendida.

Os estados mentais representados foram: crenças, desejos e intenções. Ao verificar que um desejo é possível de ser realizado o ACOMED assume então, um compromisso de realizar a tarefa solicitada e para tanto, traça os planos para atingir a meta.

Quanto às propriedades que um agente inteligente possui, podemos identificas as seguintes no ACOMED:

- Autonomia: o ACOMED tem um grau de autonomia que o permite realizar as tarefas, sem intervenção humana, pois ele possui capacidades específicas e delimitadas que permitem decidir e realizar ações a fim de atingir seu objetivo.
- Comunicabilidade: o ACOMED comunica com os demais agentes do sistema, via código.
- Cooperação: Os agentes do sistema cooperam entre si para atingirem um objetivo. Nenhum agente consegue efetuar uma tarefa individualmente.
- Reatividade: os agentes do sistema reagem respondendo aos estímulos oriundos do ambiente ou de outros agentes.
- Sociabilidade: o aspecto da cooperação é um fato que justifica a propriedade da sociabilidade, bem como a troca de conhecimento entre os agentes, interagindo entre si para alcançarem um objetivo.

- Reusabilidade: O ACOMED pode ser reutilizado para uma nova tarefa desde que o mundo onde será buscada as informações (base de dados) seja o mesmo.

Assim como no AGENTE-M, as características da linguagem LOOM permitiram a implementação do ACOMED de acordo com seu projeto, no sentido em que conseguimos criar uma estrutura distribuída em componentes e implementamos a ontologia para conceitualizar os estados mentais e as demais informações pertencentes a sua base de conhecimento.

A operação do ACOMED é bastante simples e através dos testes realizados, verificamos que o nosso agente mediador raciocina e atua corretamente ao buscar os diversos tipos de informações contidas na base, assim como na tarefa de atualiza-las.

O tratamento da comunicação entre os agentes no ambiente do ACOMED é muito fraca, embora esteja mais forte que no AGENTE-M. Esta dificuldade se justifica pelo fato da limitação da linguagem *Common Lisp*. Acreditamos que se pudéssemos ter utilizado uma versão completa da ferramenta, a comunicação entre os agentes teria sido melhor.

Podemos verificar que a construção de um módulo mediador entre aplicações e base de dados, assim como proposto por Wiederhold, é extremamente complexa. Por isso, a fim de estudo, foi apenas desenvolvido um protótipo de um agente mediador cognitivo, considerando os principais aspectos da mediação.

Finalmente, concluímos que a associação de estados mentais ao agente mediador estabelece uma íntima relação com o seu comportamento e o pensamento humano, pois ao desenvolvermos o ACOMED tivemos como preocupação a representação dos atos de uma pessoa com relação à tarefa de localizar uma determinada informação e atualiza-la. Destacamos que essa representação consiste de metáforas que simulam parcialmente o comportamento humano; uma representação perfeita está longe de ser conquistada.

Como trabalho futuro, propõe-se o estudo de agentes mediadores cognitivos em diferentes ambientes para avaliarmos o quanto este paradigma é poderoso e quanto o mesmo facilita a o acesso a base de dados em ambientes diferentes.

Propõe-se ainda a implementação do estudo de caso descrito nas seções anteriores, utilizando ferramentas atuais para banco de dados, como *Oracle* e *SQL Server* e utilizando

a versão completa do *Allegro Common Lisp*, para beneficiar-se de todas as suas características.

Referências Bibliográficas:

[ALV97] L. O. Alvares e J. S. Sichman. *Introdução aos Sistemas Multi-agentes. In*: Jornada de Atualização em Informática - 1997, Brasília, DF. Anais do Congresso Brasileiro de Computação. Brasília; Universidade de Brasília, 1997.

[ARE92] Y. Arens, C. Y. Chee, C. Hsue C. A. Knoblock. *Retrieving and Integration Data From Multiple Information Sources.* In: International Journal on Intelligent and Cooperative Information Systems (IJICIS), 1992.

[BEL95] M. Belgrave. *The Unified Agent Architecture: A White Paper*, 1995. Disponível em http://www.ee.mcgill.ca/~belmarc/uaa_paper.html .

[BRA96] J. Bradshaw. *KAoS: Na Open Agent Architecture Supporting Reuse, Interoperability and Extensibility,* Seattle, 1996.

[BRA98a] F. M. T. Brazier, C. M. Jonker, J. Treur, e N.J.E. Wijngaards. *Compositional Design of a Generic Design Agent.* In: G. Luger, L. Interrante (eds.), *Proc. of the AAAI Workshop on Artificial Intelligence and Manufacturing: State of the Art and State of Practice*, AAAI Press, 1998, pp. 30-39.

[BRA98b] F. M. T. Brazier, C. M. Jonker e J. Treur, *Principles of Compositional Multi-agent System Development.* In: J. Cuena (ed.), *Proceedings of the 15th IFIP World Computer Congress, WCC'98, Conference on Information Technology and Knowledge Systems, IT&KNOWS'98*, 1998, pp. 347-360..

[BRA99a] F.M.T Brazier, F. Cornelissen, C.M. Jonker, e J. Treur, *Compositional Specification of a Reusable Co-operative Agent Model;* International Journal of Cooperative Information Systems. In press, 1999

[BRA99b] F. M. T. Brazier e J. Treur, Compositional Modelling of Reflective Agents. *International Journal of Human-Computer Studies,* vol. 50, 1999, pp. 407-431.

[CHE85] E. Cherniak e D. McDermtt, *Introduction to Artificial Intelligence,* Addison Wesley, Massashustts, 1995, em J. M. Barreto, *Inteligencia Artificial no Limiar no Século XXII,* ρρρ edições, Florianópolis, 1997.

[COE96] M. Coen. *Sadabot Agent;* 1996. Disponível em http://www.ai.mit.edu/people/sodabot .

[DEM92] Y. Demazeau , J. Sichman e O. Boissier. *When can knowledge-based systems be called agents?* , IX Simpósio Brasileiro de Inteligência Artificial, Rio de Janeiro (RJ), 1992.

[FLE97] M. Flecha. *Processamento Semântico na Mediação de Consultas a Bases de Dados Públicas.* Dissertação de Mestrado apresentado ao Departamento de Ciência da Computação, Universidade Federal de Minas Gerais, 1997.

[FRA96] S. Franlin e A. Graesser. *Is it na Agent, or just a Program?: A taxonomy for Autonomus Agents",* University of Memphis, 1996. Disponível em: http://www.msci.memphis.edu/~franklin/AgentProg.html .

[FER91] J. Ferber e L. Gasser. *Intelligence Artificielle Distribuée.* Internacional Workshop on Expert Systems & Their Aplications, 1991.

[GEN94] J. H. Gennari at al. *Mapping Domains to Methods in Support of Reuse.* Int. J. Human-Computer Studies, 339-424, 1994.

[GIR99] L. M. M. Giraffa, *Uma arquitetura de tutor utilizando estados mentais.* Tese de doutorado apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal do Rio Grande do Sul., Porto Alegre - RS, 1999.

[GUI00] A. M. Guimarães. *Agente-M: Um Matriculador Inteligente.* Dissertação de Mestrado apresentada ao Departamento de Informática da Universidade Federal do Paraná, 2000.

[HAY99] C. C. Hayes. *Agents in a Nutschell – A Very Brief Introduction.* IEEE Transactions on Knowledge na Data Engineering – IEEE Computer Society, 1999.

[HEI95] K. Heilmann, D. Kihanya, A. Light e P. Musembya. *Intelligent Agents: A Technology and Bussiness Aplication Analysis; 1995.* Disponível em: http://haas.berkeley.edu/~heilmann/agents/index.html .

[LOOM] Ferramenta para o desenvolvimento de Sistemas Baseados em Conhecimento, construída pelo grupo de Inteligência Artificial do ISI da Universidade da Califórnia do Sul e disponível em http://www.isi.edu/isd/LOOM .

[MAC99] R. MacGregor, *LOOM Retrospective*, agosto de 1999. Grupo de Artificial do ISI da Universidade da Califórnia do Sul e disponível em http://www.isi.edu/isd/LOOM/papers/macgregor/Loom_Retrospective.htm, acessado em 27/12/99.

[MAT00] I. Mathias. *SISMAT: Sistema de Matricula Inteligente.* Dissertação de Mestrado apresentada ao Departamento de Informática da Universidade Federal do Paraná, 2000.

[MOR99] M. C. Mora, *Um Modelo de Agente Executável.* CPGCC – UFRGS, Porto Alegre, 1999.

[NOR98] P. Norving, *Paradigms of Artificial IntelligenceProgramming: Case Studies in CommomLisp*; Morgan Kaufmann Publishers, San Francisco, California, 1998.

[OLI96] F. Oliveira,Inteligência Artificial Distribuída; In: IV Escola Regional de Informática – SBC. SC, 1996.

[RIC95] E. Rich, *Artificial Intelligence;* New York; McGraw-Hill, 3ª edição; 1995.

[RIB98] M. A. C. Ribas. *Uma Solução Para a Transferência de Dados Via Internet.* 1998.

[RIV96] C. River. *Agent Technology*; 1996. Disponível em: http://www.opensesame.com/webpages/sesame/whitepaper.html.

[RUS95] R. Russel e P. Norving. *Artificial Intelligence: a modern approach.* Prentice-Hall, Englewood Cliffs, NJ, 1995.

[SAGU] Projeto de Pesquisa do Departamento de Informática da Universidade Federal do Paraná, 1999-2000.

[SCH94] G. Schreiber, B. Wielinga, R. Hoog, at al. *CommonKADS: A Comprehensive Methodology for KBS Development.* IEEE Expert, 1994.

[SIMS] Sistema para acesso a múltiplas bases de dados heterogêneas, desenvolvido pelo grupo de pesquisa em Inteligência Artificial da Universidade da Califórnia do Sul. Informações: www.isi.edu/isd/loom/projects.htm

[SHO93] Y. Shoham, Agent-Oriented Programming. Arificial Intelligence, Amsterdam, vol. 60, no. 1, 1999.

[SOU97] E. M. S. Souza. *Uma Estrutura de Agentes para Assessoria na Internet,* Dissertação de Mestrado, UFSC, PEPS, Brasil, 1997.

[TRE99] J. Treur at al, *Compositional Design and Reuse of a Generic Agent Model;. Applied Artificial Intelligence Journal,* Revised version submitted, 1999. Shorter version in: Proc. of the Knowledge Acquisition Workshop, KAW'99, Banff, 1999.

[TUT99] Tutorial para o LOOM versão 2.0.1, Grupo de Inteligencia Artificial do ISI da Universidade da Califórnia do Sul e disponível em http://www.isi.edu/isd/LOOM/documentation/.

[VIR95] S. Virdhagriswaran. *Mobile Unstructured Business Object Technology.* http://www.crystaliz.com/logicware/mubot.html, 1995.

[QUI99] A. Quintero et al. *Agentes y Sistemas Multiagente: Integración de Conceptos Básicos.* Grupo de Investigación HIDRA, Departamento de Ingeniería de Sistemas y Computación, Universidade de los Andes.

[WER95] Werneck, Vera Maria Benjamim; *Ambiente de Desenvolvimento de Sistemas Baseados em Conhecimento*; Tese de Doutorado; UFRJ; junho de 1995.

[WIE90] G. Wiederhold, P. Rathmann, B. Thierry, S. Byung, Q. Dallas. *Partioning and Composing Knowledge.* Information Systems, 1990.

[WIE92] G. Wiederhold. *Mediators in the Archiecture of Future Informations Systems.* IEEE Computer Magazine, março 1992.

[WIE95] G. Wiederhold e M. Genesereth. *The Basis for Mediation.* Maio, 1995 in Proc. International Conference on Cooperative Information Systems (COOPIS 95). Viena, 1995.

[WIN81] P. H. Winton. *Inteligência Artificial: Livros Técnicos e Científicos*, Rio de Janeiro, 1981 em J. M. Barreto, *Inteligencia Artificial no Limiar no Século XXII,* ρρρ edições, Florianópolis, 1997.

[WOO95] M. Wooldridge e N. Jennings. *Intelligent Agents: Theory and Pratice.* Knowledge Engeneering Review v.10 No. 2, United Kingdom, 1995.